

LOCALIZED MOTION CONTROL OF  
DYNAMICALLY SIMULATED ELASTIC MODELS

by

Anthony Robert Greway

B.S. Computer Science and Engineering, University of Colorado, 2006

A thesis submitted to the  
Faculty of the Graduate School of the  
University of Colorado in partial fulfillment  
of the requirements for the degree of  
Master of Science in Computer Science and Engineering

2012

**This MS Thesis for the Master of Science**

**Degree by**

**Anthony Greway**

**Has been approved**

**By**

**Thesis Advisor**

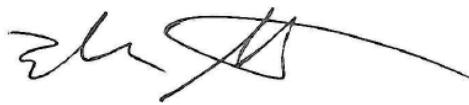
A handwritten signature in black ink, appearing to read 'M. H. Choi', written over a horizontal line.

**Professor Min-Hyung Choi**

**MS Committee**

A handwritten signature in black ink, appearing to read 'Gita Alaghband', written over a horizontal line.

**Professor Gita Alaghband**

A handwritten signature in black ink, appearing to read 'Ellen Gethner', written over a horizontal line.

**Professor Ellen Gethner**

Anthony Robert Greway (M.S., Computer Science and Engineering)

Localized Motion Control of Dynamically Simulated Elastic Models

Thesis directed by Associate Professor Min-Hyung Choi.

Physics based simulations have been flourishing for many years and provide us with an excellent method to formulate the overall motion of an elastic model. Inherently these simulations are limited by sensitive initial conditions and other material constraints that make it difficult to achieve an intended motion. Point and key-frame controls have more complete control over a deformable object and offer the ability to place a node at any position, but this can prove to be very difficult to manage for large scale animations. Artists seek perfection in an animation yet require the autonomous characteristics of a dynamics simulator. We propose a novel approach to unite the physics simulator with what we have termed a behavior based pose-metaphor. This behavior based pose-metaphor enables artistic control and creativity over a dynamically simulated deformable object without the need to change initial conditions. This control comes through direct force manipulation and localized control at sub regions of the deformable structure. In addition to metaphoric representations of elastic models, this method has instrumented state driven time control to apply metaphoric effects at specified target frames. We have implemented a few basic behavior based pose-metaphors to demonstrate our methods but new innovative user controls like touch screen gesturing and augmented reality can easily leverage and enhance this framework.

The form and content of this abstract are approved. I recommend its publication.

Approved: Min-Hyung Choi

## **DEDICATION**

I dedicate this work to my always encouraging and supporting parents Louis and Gloria Greway. Without all their love and support throughout my life I most certainly would not have achieved this goal.

## TABLE OF CONTENTS

### CHAPTER

I. INTRODUCTION .....	1
Overview.....	1
Motivation.....	2
II. RELATED WORK.....	4
Deformable Body Techniques .....	5
Elastic Models.....	9
Motion Control.....	12
Directable Animations .....	14
Current Challenges.....	16
Summary of Contributions.....	18
III. ARCHITECTURE AND DESIGN.....	19
Concept Design.....	19
Bullet Physics Engine .....	20
High-level Architecture .....	22
Linear Math.....	23
Bullet Collision and Dynamics. ....	23
Bullet Soft Body. ....	23
Bullet World.....	23
Timeline. ....	24
Base Metaphor. ....	24
Local Control. ....	24
Error Correction. ....	24
Bounding Structures.....	24

Widgets. ....	25
Local Control Design .....	25
IV. METHODS AND IMPLEMENTATION.....	27
Scrubber .....	27
Timeline .....	29
Metaphor .....	30
Soft Body Local Coordinate System.....	32
Bounding Structures.....	35
Applied Force Correction Algorithms .....	38
Force Dampening.....	39
Force Mitigation.....	41
Ease-in and Ease-Out.....	43
V. RESULTS AND DISCUSSION.....	46
Experimentation.....	46
Geometric Structures and Resting State .....	47
Moving Objects and Time Constraints .....	48
Limitations .....	50
VI. FUTURE WORK.....	52
Force Correction .....	52
Temporal Material Configuration .....	53
Example Based Implementation .....	53
User Interfaces .....	54
Augmented Reality Data Collection .....	55
Brain-Computer Interface .....	56
Validation.....	56

REFERENCES .....	58
------------------	----



## LIST OF FIGURES

### Figure

Figure II.1- Control Polygon and Deformed Grid. ....	5
Figure II.2 - FEM Diagrams. ....	7
Figure II.3 - Adjacent Control Points. ....	16
Figure III.1- Layered Architecture.....	23
Figure IV.1- Physics Scrubber.....	27
Figure IV.2- Timeline. ....	30
Figure IV.3- Localized Metaphor. ....	31
Figure IV.4- Global and Local Coordinates. ....	33
Figure IV.5- Local Coordinate System Construction. ....	34
Figure IV.6- Bounding and Adjacent Nodes. ....	37
Figure IV.7- Localized Control Artifacts.....	40
Figure IV.8- Force Mitigation.....	42
Figure IV.9- Ease Timeline.....	44
Figure V.1- Sample Experimentation. ....	46
Figure V.2- Geometric Structures.....	48
Figure V.3- Swimming Shark Experiment. ....	49
Figure VI.1- Localized Regions.....	54

## **LIST OF ABBREVIATIONS**

API	Application Programming Interface
AR	Augmented Reality
DE	Differential Equation
DOF	Degrees of Freedom
FEM	Finite Element Method
FFD	Free Form Deformation
NSF	National Science Foundation
SOFA	Simulation Framework Open Architecture
XML	Extensible Markup Language

# CHAPTER

## I. INTRODUCTION

### Overview

There are an overwhelming number of state of the art methods for simulating soft-body objects that render extremely realistic deformations in the computer graphics industry. Unfortunately there is no unified solution that can be applied to all objects or materials which give an animator the ability to easily dictate the exact motion and final results of a simulation. Many globally deterministic approaches like physics based simulations can provide us with a pivotal starting point but lack the granularity needed to offer precise articulation of a deformation. Localized control of these soft-bodies also provides us with a tool to ensure conformance of specific object poses but does not always lead to smooth and realistic animations. However, a new direction towards localized control combined with a sound global control framework looks to have promising results for hyper-realistic simulations of soft-body objects. Combining this with the ability for an animator to interact with the simulation by inputting spatial and temporal constraints at key points of an animation may lead to more desirable deformations.

This thesis aims to advance some of the fundamental problems that are currently faced in computer visualization as well as issues that extend into human computer interaction. The computer graphics field stands at the root of these challenges but they also span across many other fields. These other fields are also heavily involved with

visualization and imaging such as advanced medical rendering and computer aided training. Currently, there are methods for controlling the deformation of a soft-body ranging from total point based control to nearly complete automation through means like physics driven simulations. Each of these methods has its benefits and limitations which makes finding the balance between two quite difficult. These new methods also strive to enable new intuitive approaches for humans to interact with the simulation tools and ultimately relay their insight directly into the outcome of the simulation. We submit our ideas and techniques, which harness these technologies, in the pursuit of advancing the accuracy of deforming body representations.

### **Motivation**

As observed in most deformable body simulations, to date, there is an abundance of deterministic motion targeting from physics based or full control based frameworks. The physics based animation has been flourishing and indeed provides us with a good overall formulation for the motion of an object undergoing deformation; however, they are limited by sensitive initial conditions and other material constraints. When a rigid or deformable object undergoes a fully physics based simulation, all control points of the object must conform to the physical properties and functional constraints such as gravity and energy conservation. This leads to rapid trial and error with initial conditions and reworking constraint positions until an acceptable simulation is achieved. These types of tests do not always lead to the desired simulation due to the difficulty in estimating material properties, environmental constraints, and particle interactions

Even though numerical controls are extremely difficult to get perfect there are many cases where realistic results can be achieved. This could mean that the animator

has reached the physically accurate rendering of the object however the intended pose cannot be reached without inducing some control parameters. In essence, the animator needs more control over the simulation to produce artistic concepts and simulations. To achieve this level of control we need to be able to capture the exact moment when the behavioral effect needs to take a place and relay the animator's intentions into the deformable object. As we have learned from others work, and our own, introducing control is a difficult problem to deal with and will have side effects on adjacent structures.

As we survey the current soft body methods we are lead to believe that the deformable body is lacking in accurate motion control and could benefit from improved direct-manipulation techniques. Direct-manipulation is inherently a difficult problem because it requires intervention in the simulation and the soft body exacerbates this issue with its huge computation complexity. There are revolutionary and exciting concepts in the existing deformable object frameworks but none have found a completely unified solution that allows artistic control and produces realistic results. The deformable object needs the flexibility of direct-manipulation but must respect the need for intuitive user interfaces.

# **CHAPTER**

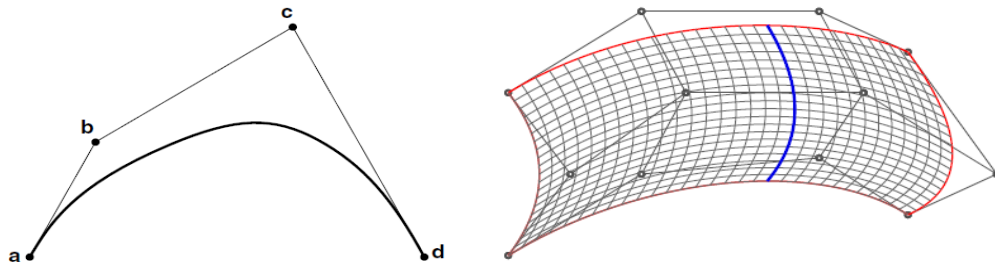
## **II. RELATED WORK**

Computer graphics has been a topic of study for many decades now. This research has elicited many fascinating and life impacting results ranging from computer aided drafting tools to life saving medical-imaging technologies. There are many different categories of graphics research even within the computing industry itself; rigid bodies, deformable meshes, cloth simulation, fluids, 3-dimensional viewing, augmented reality, and many more. Subcategories of computer graphics still challenge many researchers to find better and more realistic algorithms to suit our needs. Many of these subcategories blur boundaries and have properties which allow them to be solved with similar disciplines. For example, deformable bodies and cloth simulations both have properties such as material elasticity, internal forces, and external forces.

The focus of this thesis is the deformable body so we will take a look at some of the technologies and algorithms that currently enable the deformable body. In addition, we see that human interaction with simulations is an also an important driving force for our techniques. The trend of human computer interaction has progressed from simple punch cards, through the keyboard and mouse, to touch screen tablets, and even virtual reality. We see that it is important to have the fundamental enabling technologies which support complex geometric transformations but it is equally important that we are able to instinctively interact with the simulator so that we can properly encode our algorithms.

## Deformable Body Techniques

Deformable (or soft) bodies are objects that have the ability to change shape over time, meaning that relative distance between two reference points within the object are not fixed over time. There are also additional properties of the soft-body which, when applied, give the material different characteristics that can be described as elasticity and plasticity. Some simulations and systems have a need for fracturing and splitting of meshes however in this discussion we are focused on soft-bodies that tend to keep their shape relatively close to their initial configuration (contrary to liquids or fluids). The elasticity of a deformable object is best described by its tendency to return back to its original configuration, usually driven by internal restoring forces like the elasticity of a rubber band. Plasticity implies that when a deformable object undergoes some type of load it will permanently retain its deformed configuration until some other external force applies a new load [10, 11, 24]



**Figure II.1- Control Polygon and Deformed Grid.**

The figure on the left represented by points a, b, c, and d constructs a control polygon. To the right, a deformed grid using FFD.

Deformable bodies can be much more difficult to simulate than rigid bodies which has lead to the derivation of many specialized techniques. In the pursuit of realistic deformations in a real-time simulator, many of these techniques have been borrowed from well known physics systems while others use simple mathematical systems. Many of these techniques are quite intuitive and others utilize complex optimizations to fit the criteria needed for the computer graphics industry. We first explore a couple of the fundamental non-physical techniques used to deform a soft body.

Mathematical splines are one of the basic methods that have been used to approximate the shape and surfaces of many geometric objects and is still widely used in computer aided geometric modeling tools today. In fact there are a variety of splines which originate from Beizer curves. They work by manipulating a finite number of control points in the plane. In figure II.1 above we see there are four control points a, b, c, and d which construct a control polygon. The tangent lines at each control point are used to interpolate a polynomial of chosen degree to deform the curve in a predictable manor. The curve is used to represent the shape or surface of the object and can be efficiently computed in a real-time simulation [11].

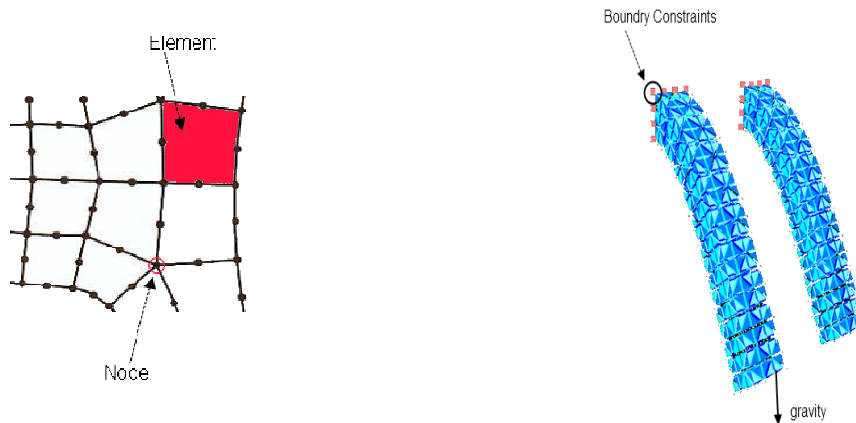
Free Form Deformation (FFD) is also a very common technique effectively used to perform primitive deformations on objects. This technique is also quite robust in nature since it can be applied atop other types of models such as points, splines, polygons, or even hybrids of these. The basic idea of a FFD is the transformation is done by a mapping function which we apply to a set of points  $p_i$  described in 3-dimensional space  $\mathbb{R}^3$ ,  $f: \mathbb{R}^3 \rightarrow \mathbb{R}^3$ . This function can be defined as the translation-rotation matrix;



where R is the proper rotation about the x, y, or z axis, and T is the transformation in space:

$$f(\mathbf{p}) = \begin{bmatrix} R_{11} & R_{21} & R_{31} & T_x \\ R_{12} & R_{22} & R_{32} & T_y \\ R_{13} & R_{23} & R_{33} & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix}$$

With a mapping function such as this we can effectively rotate, translate, skew, or stretch the model - defining points to our liking without having to adjust every control point on an individual basis. We give up some control but we can generate many more complex deformations than we could with the spline based approach alone [11].



**Figure II.2 - FEM Diagrams.**

The figure to the left is an FEM diagram showing a simple node and element. On the right is a figure of a beam undergoing FEM deformation. The red dots are boundary constraints which hold one end of the beam in place while gravity acts as an external force.

Computer based visualization has also benefited from the advances in other disciplines that have similarities in problem characteristics. In particular, the study of physics based elastic models has been borrowed and modified to fit the requirements of a

computerized deformable body. The use of physics helps to aid the simulation by supplementing many of the calculations that are necessary with governing equations and shape functions. And due to this fact, when we inject physical models into our simulations we tend to lose some of our freedom in controlling the exact outcome of the animation. We now become dependent on the methods and equations that we use to calculate the position of the points in our models.

In continuum mechanics the physical universe is viewed as a collection of deformable bodies which are made up of material points with a smooth boundary. Additionally the material points, or matter, are considered to be continuous and not discrete. This is the fundamental difference between the finite element method and continuum mechanics since we divide the body up into many finite elements. To use FEM, the body is cut up into different elements that are then reconnected back to each other with what is called a node. The nodes serve as integration points and allow the structure to pivot - giving the point a certain number of Degrees of Freedom (DOF). In a continuum the DOF are considered to be infinite since, by definition, the material is continuous while the finite element method only has a finite number.

The fundamental concept of the finite element method is contrived by differential governing equations  $L(\phi) + f = 0$  and boundary conditions  $B(\phi) + f = 0$  on the object. Since the objects or bodies that we are deforming are geometrically complex, we break them apart and form a large set of simultaneous algebraic equations  $[K]\{u\} = \{F\}$  which are used in the approximation of the finite element method. In the elastic model, matrix  $K$  contains the property of stiffness,  $u$  is the displacement of the element and  $F$  is the action of the equation - known as force. Typically in computer graphics we internalize

the system with a mass spring model so that  $K$  is a matrix containing the spring stiffness coefficients,  $u$  is the displacement of the spring, and  $F$  is the load of the spring. The connecting nodes also allow for adjacent elements to place force on each other ultimately leading to the deformation of the body [29].

In addition to the finite element method and continuum mechanics deformable bodies utilize other tools like affine frames to encode and retrieve information from the object. An affine frame is a set of vectors  $\{V_i | i = 1, \dots, N\}$  at a point  $o$ , where  $o$  is the origin of the frame and  $N$  is the dimension of the vector in affine space. Rudimentary deformations used affine frames to skew and stretch a deformable body based on the geometric relationship to the affine frames however these frames can also be useful when embedded in the body itself. When placed within the internal structure we can fix a frame in local coordinate space or let them move freely but when the body undergoes some external stress the internal frames bend and rotate with the body. This enables the control frames to measure data, providing translation and rotational information about the internal structure. The influence of each control frame in the object's internal structure can then be interloped with tools like weight functions similar to the finite element method's shape functions [12].

### **Elastic Models**

Elasticity, as used in general physics terminology also applies here in the computer graphics field. The elasticity of a material can be described as the tendency of that object to return to its original shape after undergoing some sort of external stress; meaning that it is a nonpermanent deformation. External forces are needed to induce stress on the object which leads to a distortion in the overall structure. And since the

deformation is nonpermanent, the object has another characteristic called strain which is the disposition of the particles within the body during this deformation [39, 40, 47]. Most elastic models which use some sort of physics nomenclature utilize internal forces to restore the object to its original shape however there are other models which do not require this (like key-frame animations). It is important to observe the elastic behavior of a model during the simulation since we will be manipulating these internal forces to create a more realistic looking deformation. In addition, many constraints must be met and other optimization techniques are used to smoothly interpolate the simulation.

Another related aspect to elasticity is shape retention - referred to as plasticity. The realism of an object's deformation in a simulated environment greatly depends on how well it mimics that of its real life counterpart. In nature we see that not every object observes strictly elastic behavior but rather when the external force is removed the object slowly retreats or only partially returns to its original shape. In addition to this phenomenon, realistic materials will fracture into smaller pieces when certain conditions are met. This involves the study of local discontinuities within the structure [38, 39, 46].

Elastic materials are ever abundant in our lives and can be observed in almost every object we can think of in this world, even if the elastic properties seem extremely minute. One particular work we find to be very influential in this area is the methods proposed in Example Based Elastic Models [23]. The goal is to express elastic materials in a scientific yet artistic manner such that we can manipulate and simulate solid materials in a computer-based simulation. To be more clear, the aim is to create a method in which an artist or scientist can input some specific information about an object's orientation in different configurations so that the object deforms based on the

characteristics of these configurations. An example of this input might be the rendering of a sponge in a resting position and then a second rendering of the sponge in a compressed (deformed) position. With this technique the graphics designer has the ability to create deformations of a solid in any manner that they see fit and in some cases can more accurately simulate the way an elastic material reacts to forces in a simulation. This method takes a complete artistic rendering of how the soft-body should look after deformation which differentiates it from strictly physics based computer animations that require initialization parameters for material and environmental properties. This could be seen as an advantageous to strictly physics based animations since tweaking of parameters may be extremely difficult or even improbable to achieve a desired deformation.

In this particular method [23], an example manifold built from the sample configurations of the object is used to project the elastic potential onto the real mesh. The example manifold is really a subset of the realizable manifold which is constructed by using the deformation gradient or Green strain tensor of an under-formed and deformed configuration. To create the example-manifold another step using the least squares minimization is used to find the closest strain to the realized manifold from a specific configuration. Once the example-manifold is constructed, it can be used to derive an example force that attracts the objects current configuration to the desired deformation. This is done by projecting the current configuration onto the manifold for computation of the potential elastic strain. Finally, all of the forces that interact with the elastic material are computed and placed into an implicit Euler solver and simulated on a classic physics based framework.

## Motion Control

Localized control of deformable objects involves placing constraints or introducing forces to a specific substructure of a mesh. There are many cases when a deformable body requires tweaking to achieve the optimal pose for the simulation. Using purely physics based techniques requires the modification of the initial conditions to control the outcome of the simulation however the physics framework ultimately takes over the whole simulation and will dictate the final results based on its core algorithms. Finding initial conditions for real objects is not an easy task and when it comes to translating them to a simulator it becomes even more difficult. A novel idea is to leverage the benefits of a physical simulation while still providing a way for the simulator to input localized controls to perform the intended target motions [15].

Localized *motion* control involves the ability to use a good physical simulation but introduce control mechanism at anticipated control points and at determined times to achieve the intended motion. Additionally, we want to let the simulation run in a smooth manner but still require that a node be at a specific position and time during the simulation. This also involves optimization and interpolating along the intended points through the time span of the clip. Expanding on the idea of specific control points we instead apply an intended motion to a set of affected points that all undergo a “pattern” based transformation [15]. When we use a set of control points instead of manipulating the individual points we are thus embedding what could be conceived as a hierarchy of localized control. Starting with the global mesh and then breaking the mesh down to a smaller subsection, then subdividing the subsections, and so on until we reach the

individual control points themselves. Unfortunately this requires direct interaction with each individual control point instead of the overall structure.

Motion control also allows the animator to indirectly dictate the motion of an object as it is subjected to the dynamics simulation. In particular, motion sketching [36] enables artistic input to rigid bodies by tracking an animator's movements of a real world object. These movements are discretized into segments that breakup the discontinuities observed when the body collides with other objects. Then through optimization techniques their method attempts to find proper conditions to match the intended target motion. This idea was originally intended for rigid bodies but does elicit interesting thoughts of how a user can indirectly manipulate an object in a dynamic simulation. The inverse dynamics approach can have benefits however it does not translate well to deformable bodies due to their large number of degrees of freedom and associated computation complexity.

Target driven control was then attempted in an animated fluid method by applying a driving force and gathering term to simulated smoke [9]. Smoke and fluid share similar properties to the deformable body due to their free forming shape and computation complexity thus making target driven simulations an attractive solution. Instead of attempting to find approximated conditions that can produce motions that only mimic the target pose, they use controlling forces to drive the simulation towards this configuration. Unfortunately, using this method does not always guarantee the final pose and there is no fine grained control over the simulation. Furthermore, this method does not allow us to interact with the driving forces to impose additional criteria while the simulation is underway.

## Directable Animations

One of the key limitations of physics based simulations is that the population of initial conditions does not always lead to the intended motion, so a more user refined approaches seem logical. Example based simulations [23] provided a way for the simulation's animator to input specific final position data so that the final results guarantee that pose. In essence, a full artistic rendering of specific positional poses can be preprocessed and then utilized in a method that smoothly blends the motions from a given starting position to the desired final pose. Again this is total control of the simulation which achieves the final pose however the simulation is at the mercy of the shape function during the interpolation from one stage to the next. Directable animations build upon this but allow for more user interaction by programming user defined time and space constraints.

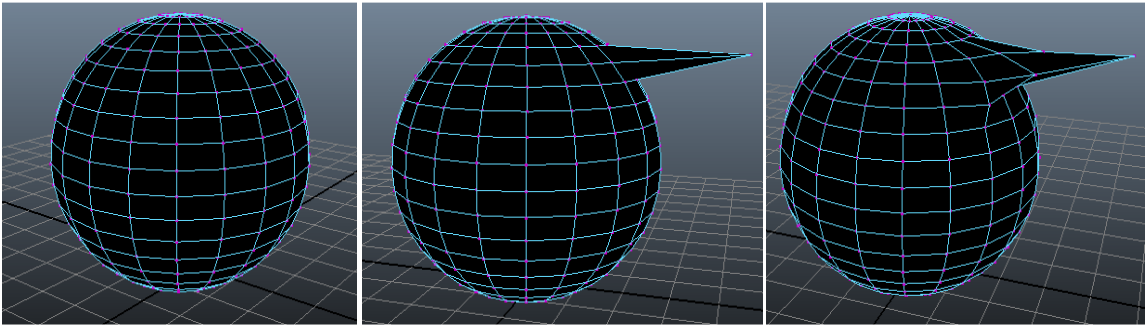
The fundamental idea behind space-time constraints is to allow for a more user definable animation of the deformable object [46]. Say that a particular object should be placed at starting position  $x_0$  but we want it move to position  $x_1$  with a certain amount of force. In essence we have a constraint problem which needs some type of optimization. We still want the animation to look realistic and satisfy a valid physical motion with regards to mass and energy so we look to an optimization function of some sorts. In 2005, Kondo et al. [18] introduced the concept of direct-able animations of elastic bodies. The animator specifies artistic key frames as an input allowing them to direct the major overall movement or deformation of an elastic object while still maintaining a plausible realistic animation. There are two techniques given in this method. First, using an FEM approach to allow rearrangement of the elastic motion but it is guided by the key frames.



The second approach modifies the objects trajectory based on the user defined input trajectory by compensating between differences. While these methods have similar characteristics to our ultimate goals they are still lacking direct manipulation of the soft body. The use of key-framing on top of the physics engine also introduces discontinuities in the simulation that can cause poor transitions between starting and stopping control forces.

In 2008, Mezger et al. [26] introduced another similar method that involves interactive physics based shape editing. This method is highly targeted at deformation of complex objects in real time but does have similar concepts to this work. Here we are dividing the substructure up in to smaller more easy to represent approximations for a speed up in algorithmic processing so that it can run at real time. There will be some loss of detail but it appears that the interactive manipulation can be realistic. Unfortunately, while the animator is able to work with a dynamically simulated object in real time they are not able to input more artistic insight other than simple pose. Many of these examples also show that this method is subject to force exaggeration in adjacent structures.

## Current Challenges



**Figure II.3 - Adjacent Control Points.**

Starting with the image to the far left we see an un-deformed object while the middle image shows an object with only a single control point. The image to the far right shows how adjacent control points could potentially be affected by the single control point.

The unification of point-based control and dynamics frameworks is an unsolved problem at this time. There are many schemes and algorithms that provide us with physics based animations and there is also an abundance of techniques which give an animator the ability to control the position of every point in the simulation. The real challenge is that neither of these two ends of the spectrum is completely correct for every type of simulation. There are many times when an animator has too much control over the simulation which leads to poor results. Imagine a framework that allows the artist to render an initial configuration and a final position mesh and then the framework interpolates between the two. We can then give the mesh some sort of physical properties, like internal resistance so the animator does not need to render every frame by hand. However we started with two mesh configurations that are ill conditioned causing poor physics results and leads to inaccurate or visibly distasteful outcomes.

This naturally leads us to the use of feedback loops for error correction to detect when objects are deforming past realistic thresholds. For example, if we observe the localized deformation of a cloth patch undergoing a curling motion at one of its corners we notice that the curling motion causes pulling of the connecting control points similar to figure II.3. Since this is a localized deformation, in which we have injected external forces, the pulling of other control points may not look visually reasonable when compared to the rest of the cloth. To cope with this, our method has deployed a feedback mechanism that identifies excessive forces in relation to the rest of the object and returns the system to a more natural order. This component is one of the key linkages between the uses of a localized control on top of a global control framework.

Relaying information concerning how to deform an object into the simulator is also a daunting task which is closely related to other challenges. Producing an algorithm or framework which can output a smooth-realistic deformation is not enough to solve all of the soft-body challenges. The human mind ultimately houses the information of how we would like to see a deformation take place as well as the criteria for how the end results are verified. Providing a means for which artists and animators can intuitively input accurate information about a deformation is just as important as the technologies that enable the simulation. Current techniques simply do not provide programmers of these simulations with the ability to easily translate their thoughts into the algorithms. We intuitively know that certain parts of an object should be in a certain configurations based on our perceptions of reality but the current tools limit our abilities to relay this information into the procedures that are driving our simulators.

## **Summary of Contributions**

In this thesis we will present the four main components that we have contributed to the computerized visualization of deformable objects. (1) The development of a framework that unifies key-frame animation techniques with dynamically driven elastic models. (2) Fine grained level of control through localized motion control. (3) Force correction techniques that recognize and correct visually inaccurate deformations (these techniques bridge the gap between the global and localized control points). (4) And finally, embed spatial and temporal constraints directly into a dynamics simulator in a user intuitive design.

## **CHAPTER**

### **III. ARCHITECTURE AND DESIGN**

#### **Concept Design**

To achieve realistic animations and even in some cases physically impossible animations we must carefully look at the current computer graphics tools at hand. While an animator, game developer, or film studio artist may have many different requirements for their final products, they all look to create the most realistic effects with the least amount of manual interaction. In most cases, a desired animation will mimic a real world object moving through a physics based world while undergoing many force changes. In all use cases, even if the graphics simulator provided results which precisely mimic the real world physical behavior this would not be enough. If we examine a computer graphics artist working on an entertainment film we see that their requirements would include the need to modify an animation in ways that violates real world physics. Most action or fictitious feature films rely on computer generated images to obtain results that simply cannot be achieved in the real world because they are physically impossible.

Physics and dynamics simulators have achieved many realistic results and even some desirable un-realistic animations. However, we recognize that while the dynamics simulator provides us with physically realistic results it comes at a cost. Dynamics simulations are deterministic and will always provide the same results when the same initial conditions and forces are applied. This requires that the animator spend a large quantity of effort dialing in the initial conditions of all physical properties in the scene. In addition to this, the initial conditions and material properties that would achieve the

desired results cannot always be applied to the entire scene or object. In terms of stiffness, if we reduce the spring dampening of an entire object to achieve a more floppy or pliable object we may see adverse side effects in other portions of the object as it moves through the physics based world.

These limitations may also lead an animator to seek out temporal constraints in a dynamics simulation. As a soft body moves through space we may not want to see this object become pliable until it reaches a specific point in time. This cannot be achieved by simply modifying and testing initial conditions in a reasonable way for the animator. Furthermore, this desired motion or characteristic may not be applicable to the entire object meaning that the animation truly requires a localized deformation. With these requirements in mind we will describe a novel design that brings spatial and temporal requirements together in a physics engine, while providing the user with intuitive localized controls.

### **Bullet Physics Engine**

Many of the principal goals of this research revolve around finding the best way to balance an autonomous deformable body simulation with user interactive controls. To best suit these needs a couple different physics engines were used to develop test applications for our experimentation. The SOFA framework was the original target platform for the simulator development based on high level investigation and documentation however during the early development stages the Bullet Physics engine was also tested. After much prototyping the Bullet Physics Engine was chosen for the final production due to its more intuitive programming interfaces. SOFA does have a

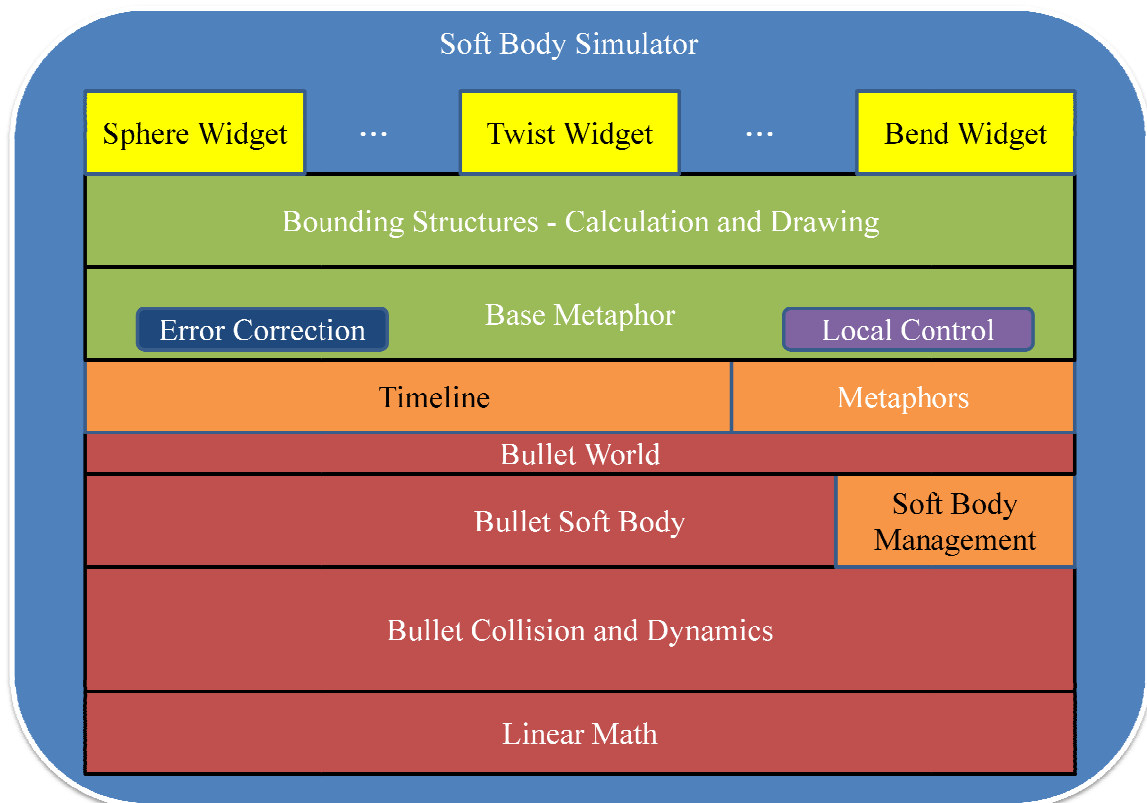
large set of resources, solvers, and plug in support but ultimately the programming interfaces did not suit the requirements of this project.

The Bullet physics engine was created by Erwin Coumans and was originally designed for gaming and film production based around rag doll physics. The project is open source and provides all the fundamental elements needed for computer graphics related research. At its core the Bullet engine is designed to simulate real time computer animations so it contains many approximations in the actual physics and in some cases defaults to Verlet type integration solvers. For this research we focus on the deformable soft body which has more complex characteristics than rigid bodies when simulating collisions and motion.

In Bullet, the soft body objects rely on many tools available in most computer graphics libraries like 3D meshing, particle based dynamics, collision detection, and mass spring-dampeners. These will play a crucial part in our implementation since it will also utilize all of these components. The meshes used in our simulator are based on simple tetrahedrons connected by particle links which are subject to spring dampening. In general, meshing geometry is often simplified in a real-time physics simulations due to the increased speed and efficiency. In our implementation convex cluster objects are used to simulate quick and efficient collision detection with the ground and the internal structure of the soft body. Using direct mesh to mesh collision algorithms are quite costly and also accumulate inaccuracies in their detection of object penetration. It is worthy to note that these meshing methods, along with many other graphics methods, inject a large measure of assumption into the final motion of the object through the simulation.

## High-level Architecture

The experimentation of this thesis was all carried out on top of the Bullet physics engine and in order to get a better understanding a high-level architectural overview is given here. We will then dive into a deeper discussion of how this architecture was implemented. For the purposes of clarity the overall framework implementation will be referred to as the *Soft Body Simulator* since at its core we have a physics engine driving soft bodies. This a layered architecture where each component builds on top of previous components so we will describe the architecture of this framework by starting at the bottom most fundamental layers and working our way up.





### **Figure III.1- Layered Architecture.**

This is a representation of the final implementation's architecture in a layered format. Starting with the bottom Linear Math layer, each component utilizes the ones below it to perform its designated tasks. The red shaded components are components of the Bullet framework and all others are custom components of the *Soft Body Simulator*.

**Linear Math.** Bullet provides an efficient linear math library that supports many of the operations required to calculate localized deformations, bounding structures, and force directions. This library provides the user with the ability to easily calculate vector operations, quaternion transformations, matrix algebra and many other geometric facilities. This library was used heavily in the construction of the bound structures, widget controls, and force correction algorithms.

**Bullet Collision and Dynamics.** This is the core of the Bullet runtime engine providing rigid body and simple body dynamics. Collision detection and other generic constraint solvers are also lumped into this portion of the library.

**Bullet Soft Body.** Soft body dynamics is quite different from the rigid body dynamics and is much more complex due to the interaction of the body with itself and the world. The structure of these objects is always changing and therefore the simulator must also take into account internal forces of the object. The timeline construction has a direct dependency on the soft body structures moving through the Bullet world and thus the *Soft Body Management* was built to facilitate the population of the physics timeline. Manipulation of the private data members required that this extension was built within the soft body.

**Bullet World.** This is implemented by the bullet engine and houses all the characters that move throughout the scene and houses global parameters like gravity and camera controls.

**Timeline.** The timeline encompasses two critical components of this implementation that provide the user with intuitive temporal control over the simulation. One is the dynamics scrubber which moves the soft body through a recorded timeline. The second critical component is the tracking of metaphors and activation for each time step of the dynamics solver. This requires the storage of direct references to different metaphor instances.

**Base Metaphor.** The construction of this class was designed to be abstract so that all widgets can benefit from unified force correction techniques but still provide enough flexibility to allow the overriding classes complete force control. In essence, the overriding classes are allowed to apply customized forces at each time step but are then subject to error correction before the differential equation solver is asked to step the simulation.

**Local Control.** The local control object is a data structure which stores all the relevant information that a metaphor needs. This includes bounding structures, soft body references, and time step information.

**Error Correction.** Error correction is built into the base metaphor and serves as housing for new and existing algorithms to act on the controlling forces before they are realized by the simulator. There are currently three different error correction techniques used to smooth and control the localized deformation forces applied by the different widgets.

**Bounding Structures.** Visualization tools are a key focus of the *Soft Body Simulator* and this component houses the implementation for most of the tools seen in the simulator. There is a high degree of vector math and linear transformations represented

in this library. This library is also responsible for the calculation of bounding areas, node adjacency lists, and the soft body local-coordinate system.

**Widgets.** Currently there are three widget implementations - each has unique characteristics for controlling the localized deformation. Widgets are a way to apply fundamental operations that computer graphics artist would need to create a customized soft body deformation. We believe that a more generalized widget that allows for more precise final configurations could aid or replace many of these implementation. However each of the existing and potentially new widgets can extend from the base metaphor and benefit from the local control structures and error correction algorithms.

### **Local Control Design**

One of the critical aspects of this study is the focus on localized control and interaction with the soft body as it travels through time. Localized control has the advantage of manipulating specific aspects of an object without destroying the overall look and feel of the animation. We feel that the amount of user defined interaction in the simulation should be as autonomous as possible to avoid large amounts of tweaking. To avoid global parameter tweaking and key frame like simulations we introduce an interactive bounding structure which the animator can directly place in a scene and receive real time feedback of their localized node selection. These localized bounded nodes and even the bounding structures themselves will follow the object in a localized coordinate system. These bounding structures ultimately dictate the force magnitude and direction based on the metaphors application.

For this research and experimentation we have chosen to stay in the force domain of the physics simulation to avoid jerky motions that result from large or instantaneous

deltas in position. The Bullet engine provides many types of Ordinary Differential Equation (ODE) solvers and in particular the bullet soft body implementation provides a Verlet solver as its default. Verlet is another numerical method used to integrate Newton's equations of motion and is frequently used to calculate trajectories of particles in molecular dynamics and of course physics simulators. The Verlet algorithm reduces the level of errors introduced into the integration by calculating the position of the next time step from the positions at the previous and current time steps (without using the velocity). It is derived by writing two Taylor expansion of the position vector in different time directions.

Basic Verlet:

$$y(t_n + h) = y(t_n) + hy'(t_n) + \frac{h^2}{2} y''(t_n) + \frac{h^3}{6} y'''(t_n) + O(h^4)$$

$$y(t_n - h) = y(t_n) - hy'(t_n) + \frac{h^2}{2} y''(t_n) - \frac{h^3}{6} y'''(t_n) + O(h^4)$$

Add the two expansions to get:

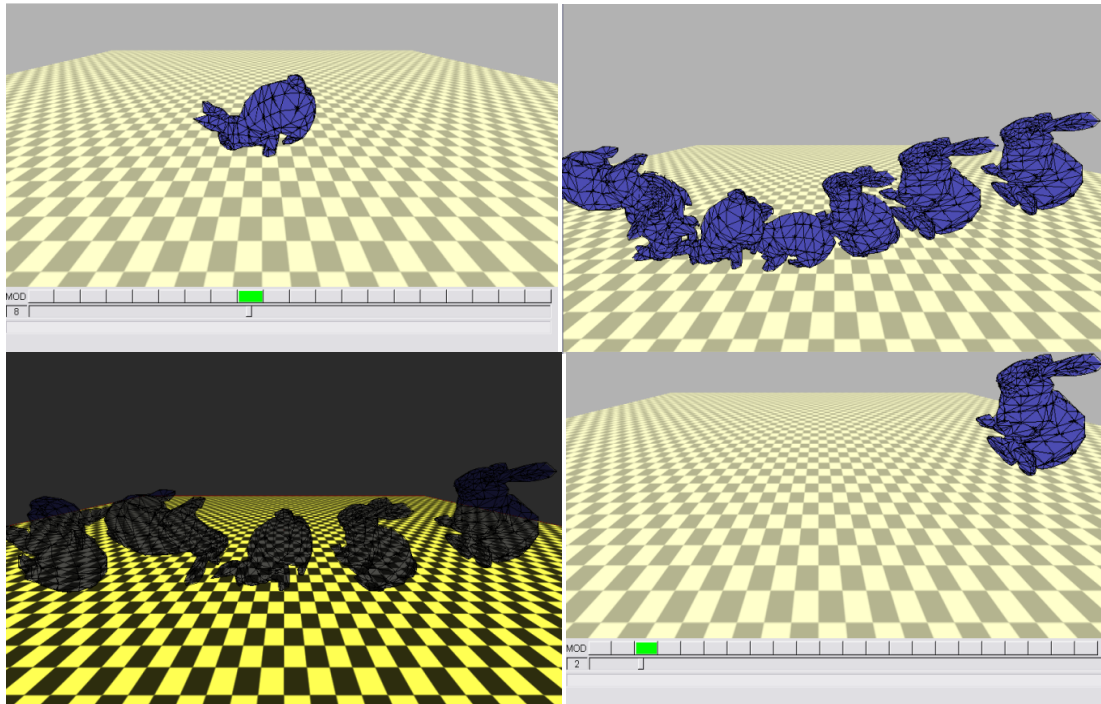
$$y(t_n + h) = 2y(t_n) - y(t_n - h) + h^2 y''(t_n) + O(h^4)$$

This removes the velocity of the particle from the calculation leaving us with only acceleration to apply. Thus we have chosen to focus on direct force interaction of the moving body to provide the best stability in our framework.

## CHAPTER

### IV. METHODS AND IMPLEMENTATION

#### Scrubber



**Figure IV.1- Physics Scrubber.**

A tetrahedron bunny moving through a dynamically stepped world - based on stored position, velocity, and acceleration. The timeline scrubber can be seen at the bottom of select images above. As you move the scrubber back and forth you can see the state of the bunny change with respect to time and at any point on the timeline you can add user defined controls.

Time constraints and motion capture are tightly integrated into this framework and provide the basis for user interaction with the target object as it is subjected to the dynamics simulation. Physics simulations are deterministic and provide us with a foundation for consistently tracking an object as it experiences external forces and motion. For this reason the animator will dial down the physics based simulation by

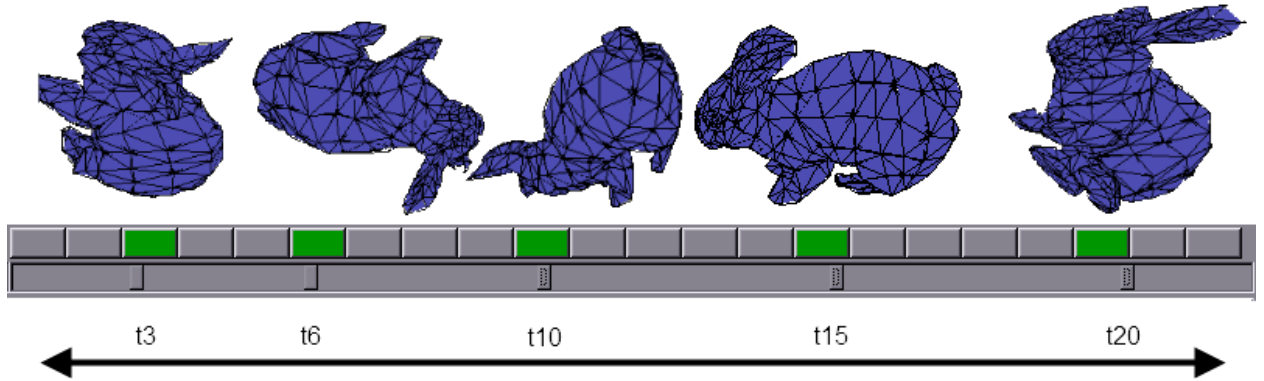
doing some minimal material property adjustments and adding force constraints to the projectile until they find a reasonable base animation. Once the scene has been initialized a recording of this motion is taken and analyzed for its legitimacy and placement in a larger production or scene. It would be a very rare case that the animation is exactly to the animator's liking and integration with other larger portions of a scene may require small modifications to the objects behavior. The animator could try to reproduce the desired results with more material or scene parameter alterations however this could completely destroy the animation or final configuration. It would be advantageous to step backwards in the physics simulation and select the precise moment the object's behavior needs to be modified. And furthermore the need for only modifying localized portions of the object to avoid jolts or unpleasant side effects is appearance.

The physics timeline scrubber is analogous to many other medium like video or audio scrubbing so the ideology has been barrowed in the same respects here. Much like you would like to skip to a specific chapter of a movie and then fast forward or rewind to an exact moment using time lapse scrubbing capabilities, the physics timeline scrubber provides the same functionality. Without destroying or perturbing the time stepped physics simulation one must live capture specific states of a dynamics body as it moves through a physics world. Once this is achieved one could scrub backwards and forwards through a dynamics simulation. Since most (if not all) simulators use approximating differential equation solvers that introduce error through calculations and storage limitations it is not practical to try and reconstruct previous time steps through negative deltas. Therefore state capture has been used in the implementation of the timeline scrubber.

The soft body management is a facility that has been built directly into the Bullet soft body implementation to allow us to save off the live internal state of the object. Through the use of indexing, copy constructors, and pointer reassignment the crucial data of the internal state can be saved off into a heap structure. Later we can use this information to directly re-step the simulation from any recording point forward. The Bullet soft body object was never designed in this manor and therefore there are certain limitations in this implementation that prevent one from using an arbitrary state for collision detection. The initial creation of the convex clusters is based on a resting configuration which implies that if regenerated in a deformed state, undesired artifacts will be introduced. In our scenario we will not be using arbitrary state to generate our clusters but rather the initial state so we can still achieve the accuracy desired.

### **Timeline**

In order for an animator to truly harness control of the animation they need to be able to manipulate the timeline at each step of the physics simulation. The recording capabilities of the timeline also allows the user to interact with the target object and physics based world in any way they see fit, whether that's adjusting the material properties or adding forces at specific configurations. The timeline's purpose is to store not only the target object's state but the position of the deformed local coordinate system, user applied metaphors, and any error correction information needed at each time step. The "Metaphor" data structure contains all this information so that specific localized controls can be applied at any time step of the simulation. In addition, one of the key features of the timeline is that it can be constructed at any point in the dynamics simulation by simply pressing the record button.



**Figure IV.2- Timeline.**

This is a visual representation of how the time line is constructed with respect to the physical state of the target object. In this representation the bunny is moving through the world from left to right as it bounces off the ground. The scrubber is highlighted at each time step that is drawn for the different time values.

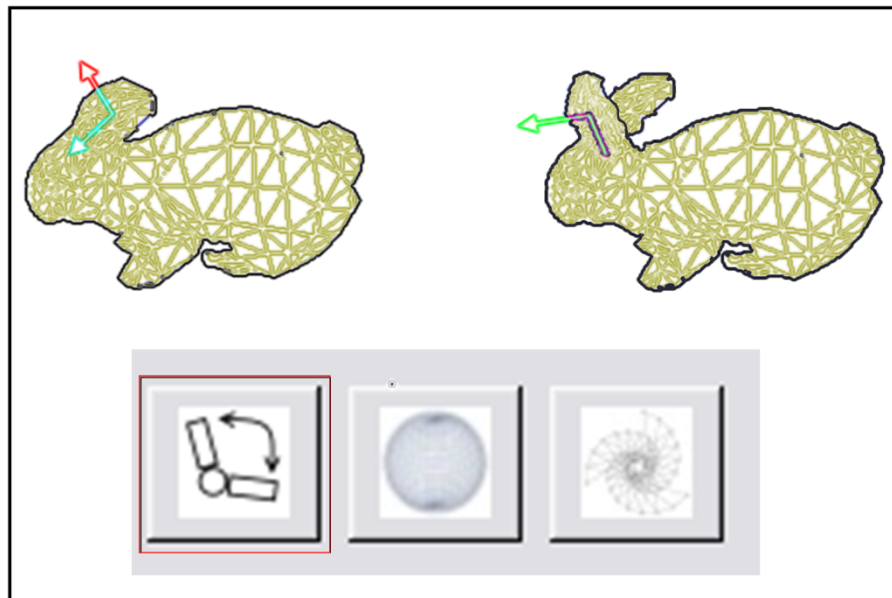
The actual data of the timeline is stored in a vector and has direct pointers back to the metaphors that would be applied at each time step. If no time step is applied at a particular dynamics step then there will be a null pointer. During the timeline preparation a copy of the initial soft body is created so that the original state is not destroyed. Again this enables the animator to deterministically replay the physics animation by applying and removing user defined local controls until the exact representation is found.

### **Metaphor**

According to the Merriam-Webster dictionary the word *metaphor* means; “a figure of speech in which a word or phrase literally denoting one kind of object or idea is used in place of another to suggest a likeness or analogy between them.” We directly apply this meaning to our intentions for the localized control of a deformable soft body. In this implementation we use widgets which we define as sphere, twist, bend, flatten, squash, and stretch. These widgets, metaphorically speaking, actually can produce edge



rounding, deflection, or other indirect motions than what the name portrays. This is because applying a control force to an object in a localized manner leaves great flexibility as to how the smaller region of the object interacts with the overall structure. For example, the flexible ear of a bunny moving through space may exhibit natural flopping, twisting, or bending however when we apply a bending metaphor we may actually see a linear projection since the base of the ear is anchored in a much larger mass structure. In a similar respect, if we use the spherical widget on the corner of a square shape we will actually see a rounding of the edge instead of a direct mapping to a sphere. We see these indirect motions and specific pose behaviors thus we have termed this the behavior based pose-metaphor.



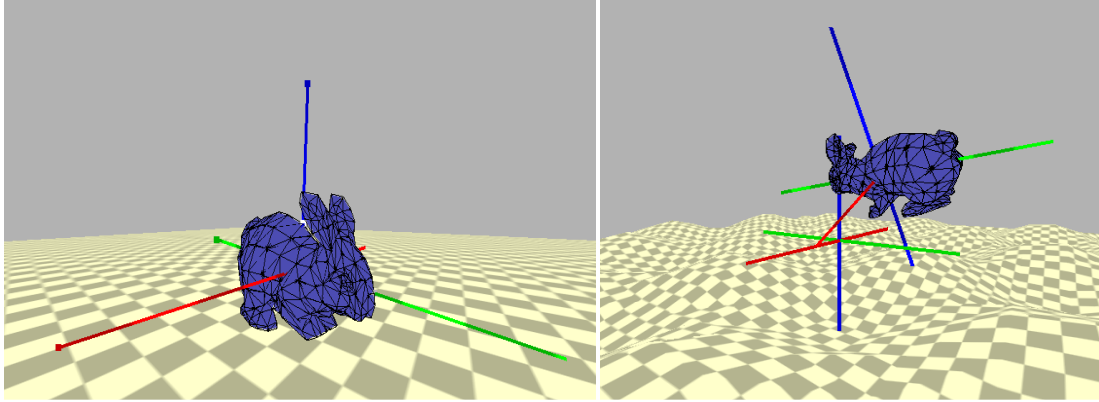
**Figure IV.3- Localized Metaphor.**

This is an artistic representation of the metaphoric properties of the bending widget. Notice that the bend metaphor has the ability to bend a flat surface in a parabolic fashion however when applied in this configuration the base structure holds one end of the localized control in place creating a linear projection. The bunny on the left is in the rest configuration with intended forces represented by the red arrows. The actual motion of the bunny is depicted on the right with the green arrow.

The behavior based pose-metaphor is one of the most critical structures of this framework acting as the linkage between the stepped simulation and the users intended interactions. An abstract class representing the *Metaphor* is constructed in an extensible way to allow new metaphors to be created as the framework is built up. We have created some primitive metaphors to represent the capabilities of localized control in a dynamics environment however there is much more work that can be implemented here in the future. Later, we will discuss in more detail how we envision the *example manifold* as a possible enhancement or replacement for all widgets and metaphors in future implementations.

### **Soft Body Local Coordinate System**

Graphical transformations and rotations are an essential part of constructing any system that manipulates objects that require interaction or collision. In a typical simulator we have the global world coordinate system which defines where all objects in a scene are placed and usual some form of a localized coordinate system for each object in the scene. In Bullet and other physics engines, rigid bodies come inherently with a localized coordinate system which can easily be tracked throughout the simulation due to consistent reference points in the body. The same cannot be said for a deformable body which has a constantly changing exterior and interior structure. This was a challenge in this implementation and we present the methodology used to construct a stable localized coordinate system on top of the physics simulator.



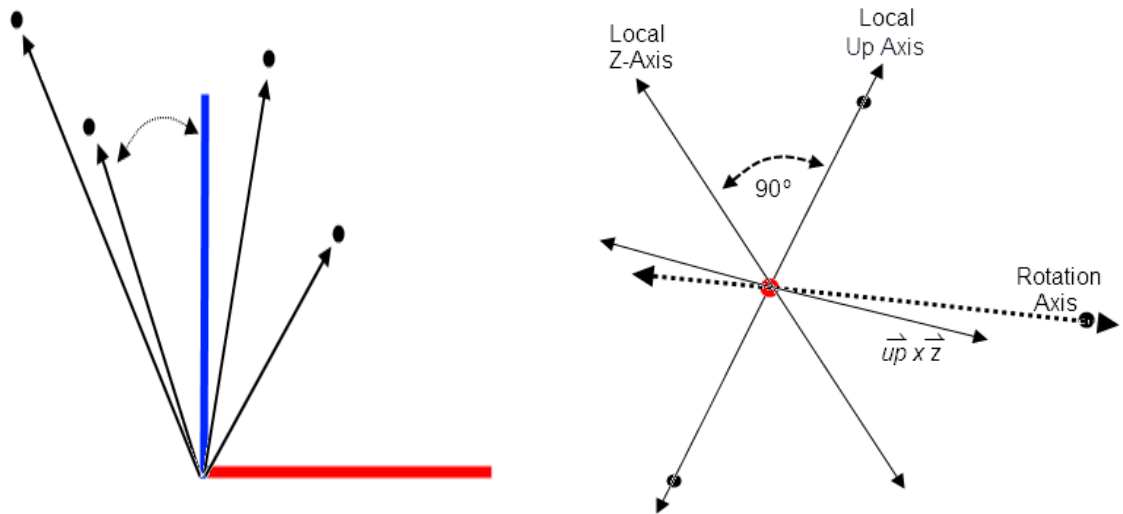
**Figure IV.4- Global and Local Coordinates.**

On the left we see a local coordinate system that represents a single soft body object. On the right, the global coordinate system is rendered in addition to the local coordinate system to better illustrate their purposes.

Since not all soft bodies are created equal and each has unique properties related to its overall structure, some assumptions were made about its orientation during initial construction. This is not a problem with rigid bodies since the nodes always stay in the same position with respect to each other and enable the local coordinate system to be easily constructed in a multitude of ways. In the case of a soft body, reference nodes should be chosen with some care to ensure that maximum rotation detection and overall stability of the coordinate system. The above example images depict a bunny with a local coordinate system found at nearly the center of mass which is much more reliable than picking somewhere near the ears or feet.

For our local coordinate system construction we assume that the object is first aligned with the global coordinate system having a large portion of the center of mass on the upward axis. Since the center of the object can be quite unstable we cannot use it as a reference point and must instead find three other reference nodes in the soft body structure to calculate the upward direction and rotation. The second step to constructing

the upward axis is to take the node closest to the global upward axis in the positive direction by calculating each node's vector normals from the origin and then finding the point closest to normalized upward global axis. The same approach is used for the bottom of the soft body. Finally a vector constructed from the bottom node to the top node represents the localized upward axis (dividing the length of this vector in half computes the origin of the local coordinate system). For rotation, we then find a third reference node on the body by searching for a maximum node in the x or z direction of the global coordinate system. However this reference node will not be used to directly compute an orthogonal axis since we cannot guarantee it will always be 90 degrees from the upwards axis. Instead, this vector represents a rotation axis. Rotating the upward axis 90 degrees about origin around this rotation axis gives us our second reference axis. Finally a simple cross product of these two vectors produces the final reference axis.



**Figure IV.5- Local Coordinate System Construction.**

The image on the left represents the location of reference nodes for the upward axis from the origin. Each of these vectors will be normalized and their distance from the global upward axis will be analyzed. On the right, we have the fully formed local coordinate system. The black dots represent reference nodes, the red dot is the origin, the

dotted line is the rotation axis, and the solid lines are the referenced local coordinate system.

We note that this is a fairly elementary approach to constructing the system and could be expanded in the future. A much more sophisticated initialization system could be utilized that eliminates the assumptions we have put into place. In particular, a torus comes to mind as a difficult soft body to construct a localized coordinate system since there is no mass at the true origin of the object.

### **Bounding Structures**

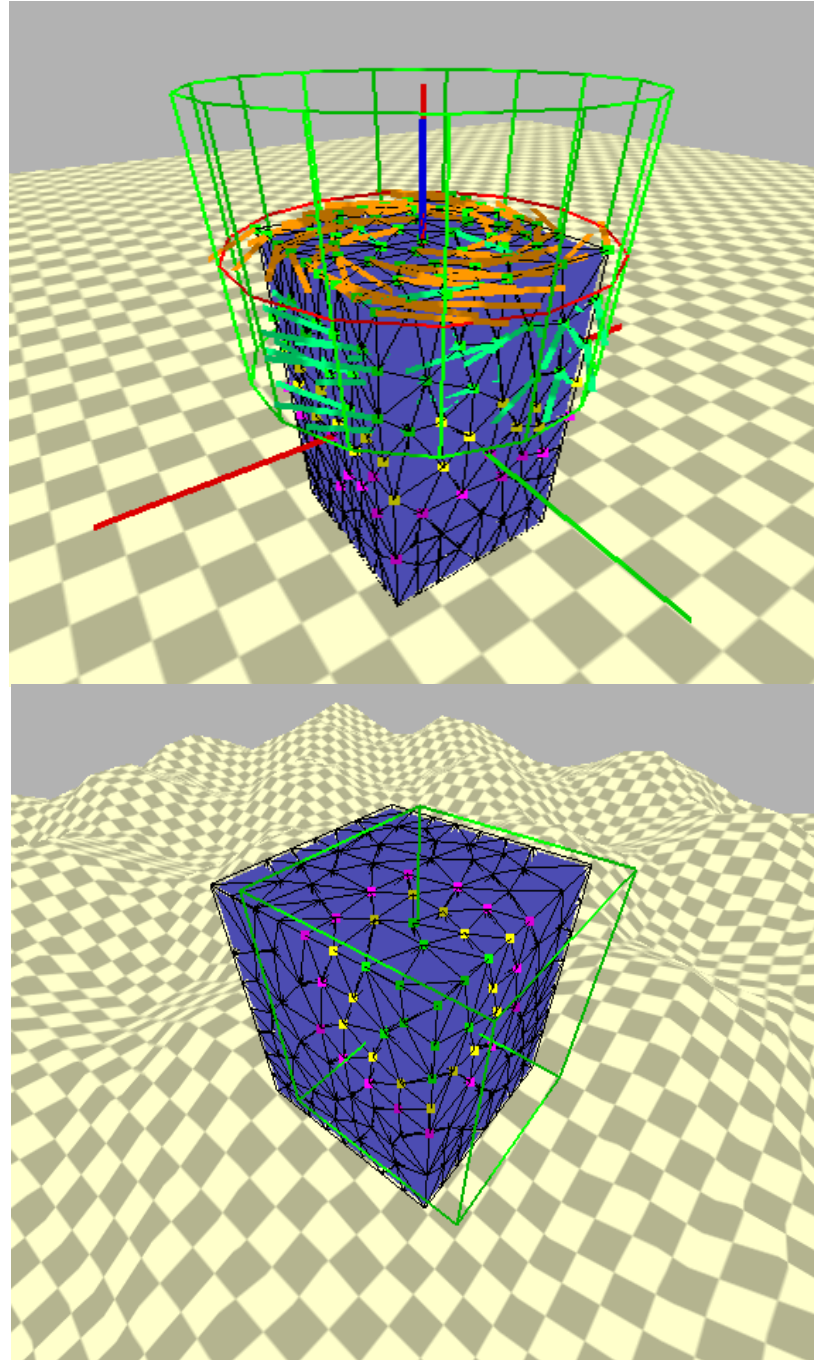
Another chief goal of this work is to find ways for an animator to interact with the scene and target structures in intuitive and concise ways. We foresee a huge potential for new and innovative ways that an artist can communicate their thoughts in the graphics algorithms. For this release of the *Soft Body Simulator* the focus was on interweaving direct user controls and the automation that comes from dynamic simulations. To perform these tasks in an instinctive manner we constructed three different bounding structures to help the animator arrange their thoughts into the simulation. Each of the bounding structures performs a unique task that directly correlates to their corresponding metaphors. For instance the cylindrical bounding structure performs a twisting type motion by producing a gradient in direct correspondence to the radius and upward axis of the bounding structure. Potential force directions are also supplied in a graphical fashion for certain bounding structure to give visualized aid to the artist.

Each bounding structure was also constructed in a manner that allows them to follow the localized coordinate system which in pinnacle to the accurate enforcement of its directed forces. Linkage of the bounding structure and the artist's vision will be an

important factor in the future work that will likely come from localized deformations.

Many artists do not think in a linear way that bounds them to simple geometric structures so it was important to find a methodology that allows customized and potentially shape-shifting bounding structures. This also brings to light ideas of static force fields that are placed directly in the path of the moving object which may require linkage between the local and global coordinate systems.

The bounding structure also performs another critical task. Error correction and manipulation of specific nodes requires knowledge of adjacent linkages in the target object. By calculating nodes that lie directly within the bounding planes and surfaces we can easily test the soft body links to see which nodes form an adjacency. The adjacency lists are calculated up to two levels and are represented in the diagrams below by the yellow (level1) and purple (level2) colored nodes. For fast lookup the lists are stored in a hash map with their parent index values as the keys.



**Figure IV.6- Bounding and Adjacent Nodes.**

The light green cylinder and cube are interactive bounding structures controllable by the animator. Green dots are nodes directly effected by the metaphor's localized force manipulation, yellow are the effected nodes immediate neighbors, and purple dots are the second level adjacent nodes. A visual representation of the intended forces on the effected nodes is show in the left diagram. The green direction lines are on the bottom portion of the cylinder and are calculated based on gradient. The orange directed lines are nodes that lie above the center plane of the cylinder and twist in the oposite direction.

## Applied Force Correction Algorithms

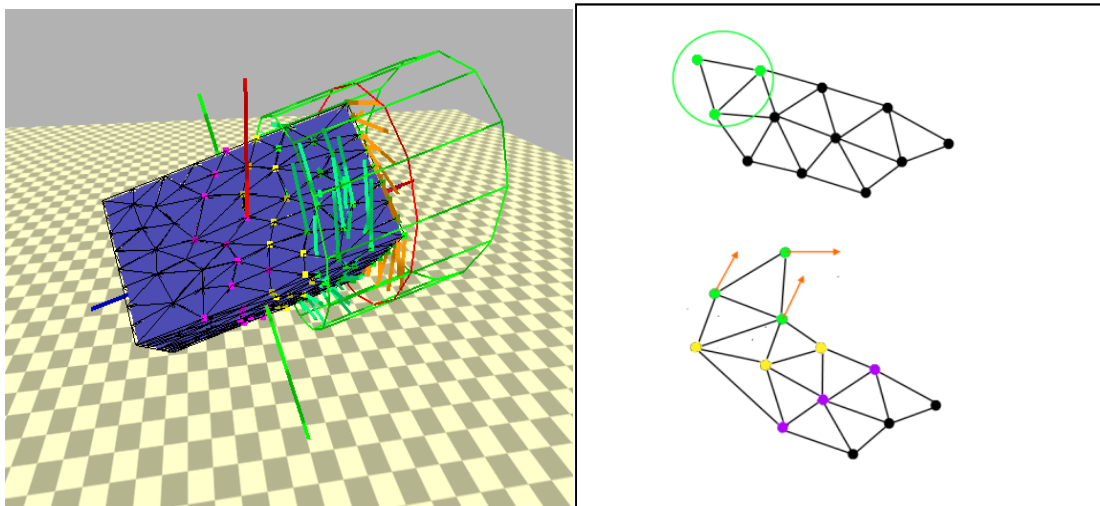
Applying forces to a deformable body or cloth like structure will directly result in change in that objects external appearance and internal force state. Because these structures are constructed by particles that are connected by some form of linkage we must be careful to preserve the overall external appearance as the object undergoes any type of induced transformation. Adding localized forces creates an interesting situation that requires the simulator to not only delegate forces to specified nodes but also deal with the side effects that can occur in the adjacent structure. Moving a single point on a deformable body can and will have a cascading effect on all the other nodes in the structure. This is apparent to us when we push or pull on a soft body in our real physical world and watch as the malleable structure reforms itself into another configuration. However, this is less apparent and in many cases undesirable when dealing with only a localized portion of a simulated soft body.

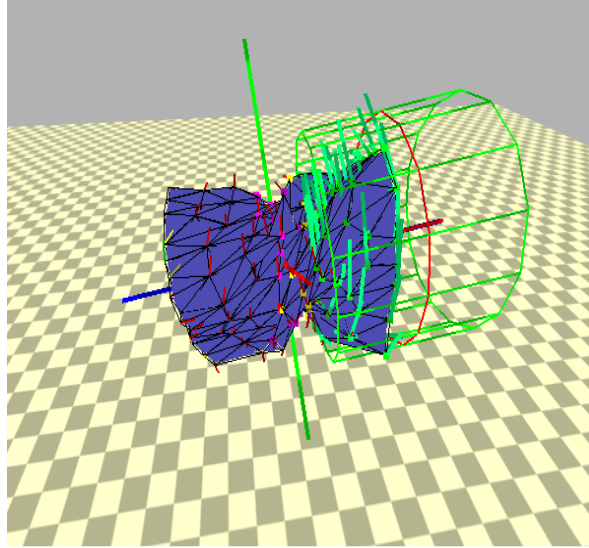
The final goal of applying localized control is to achieve the animator's vision within the computerized simulation. We see these correction algorithms as tools which enable more complex gestures and provide a mechanism for the *Soft Body Simulator* to further refine the applied forces. Currently we do not have a human computer interface that allows the artist to pick and pull at the structure with their own hands but imagine, if you will, an input device that places a real object in your hands and reads your interactions with it. With one hand you hold the base of the object and with the other you curl a flexible limb upwards. How would this be translated into an algorithm that only moves the flexible limb in a natural way while not perturbing the overall structure? Without a full two hands haptic interface the user interface will always require some



transformation from one mode to the other. Here we describe three different approaches to controlling these forces which are used to aid in some of these gestures.

**Force Dampening.** A simplistic but effective approach to resolving unwanted behaviors introduced by localized control is to apply force dampening directly to adjacent nodes in the soft body structure. The bounding structure describes the nodes that the artist would like to be affected by the control metaphor but blindly applying forces to these nodes is not enough to achieve a smooth deformation. Through trial and error the artist does have some control by increasing or decreasing the bounding area of effected nodes which will directly change the outcome. However there are situations that cannot be corrected by this level of control. For these cases we impose a more intuitive approach that allows the artist to enable or disable adjacent node force dampening. The dampening provided by this algorithm is imposed on proceeding steps of the dynamic simulation after localized forces have been applied.





**Figure IV.7- Localized Control Artifacts.**

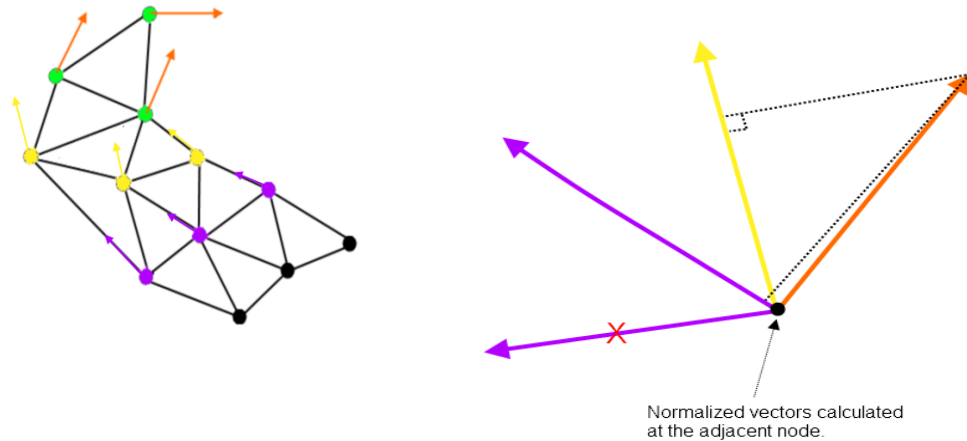
The examples shown in IV.7 give us a visual example of unwanted artifacts in a twisting or curling motion of a simple mesh structure. As the green nodes are subjected to the localized control forces they pull on the L1 and L2 adjacent nodes causing large amounts of stretch and load on the linking structures. In particular the bottom-left corner of the mesh is undergoing extreme deformation and large amounts of load are being placed on the adjacent node linkages. In some cases this may be desirable but if the forces were untamed and left to continue on their course there is a potential for them to completely wrap around and even cause a collision with the body itself. We believe that in many cases this is not the desired effect therefore we can impose a linearly-decreasing dampening of forces on the adjacent nodes. There is some loss of energy in node linkages and internal energy as you traverse away from the affected nodes but this is not enough to counter act the applied forces. We directly impose a linear dampening of 50 percent on the L1 nodes and 25 percent loss on the L2 nodes. This is a strict loss of

energy and is completely agnostic to the forces being applied through the localized bounding structure.

We have found through testing and experimentation that even though this is a fairly crude form of stabilization it does provide the simulation with a more controlled and smoothing effect. This is similar to a semi stiff anchor being placed at the point of force contact somewhat like a hand holding one end of a soft body as the other applies force. One of the unfortunate side effects of this dampening is that all components of the force vector are affected by the diminish magnitude - including that of gravity. Depending on the scene and situation this type of loss can go undetected by the human eye particularly when external forces are being applied. In an attempt to perfect this dampening we have sought out a more insightful algorithm that peeks into the localized control structure and directly interacts with the forces being applied.

**Force Mitigation.** Force mitigation has been named so due to its ability to mitigate the effects of the applied local controls as they spread into the deformable structure. By having knowledge of the applied forces we can make more sophisticated assumptions about how to better dampen the forces before causing unwanted artifacts in the deformation. This particular type of force correction allows the body to move in less restricted motions through the physics world and more in the natural direction of the bend or twist caused by the localized control. Again this may or may not be the desired effect that the artist is looking for therefore this parameter is configurable for the animator to enable and disable as they please. There is a somewhat subtle difference in visual effect with this algorithm when compared to the brute force dampening however it does provide a different visualization outcome.

A key principle to this algorithm is that it injects itself directly into the metaphor and requires that the metaphor writer relay the force direction at each time step for it to function properly. At its core, this algorithm first checks that the adjacent node's force direction is comparable to the applied force direction by using a simple vector dot product. When adjacent nodes do *not* fit this criteria they, and potentially their descendants, are skipped due to a lack of force contributing to their current motion. Because we are keeping the adjacent nodes in a hash map where the effected node is the hash key, we must keep in mind that nodes can have multiple adjacencies. To truly mitigate the force based on direction we must take into account the force direction of each of these parent nodes in the descendant's adjacencies. Each adjacency is mitigated on an individual basis and a diminished force dampening occurs when we compound force projections for all its parent nodes.



**Figure IV.8- Force Mitigation.**

Directional force vectors have been added to the simple meshing structure at the L1 and L2 adjacency nodes to help to visualize the force projections.

The green nodes are nodes affected directly by the localized control forces and all participate as parents of force mitigation except for the topmost node since it does not have direct descendents. As we compare the dot products and produce projection vectors we see that one of the L2 nodes is not mitigated while the others show at least some of their current velocity is due to the introduced forces. We assume that a force vector with a dot product less than zero has little or no contributing forces from the localized control. Now that we have obtained the projection vectors due to applied forces, we subtract a portion of that contribution from the original force direction in the adjacent node. This will alter the vectors direction and magnitude to help mitigate the changes due to the applied forces.

**Ease-in and Ease-Out.** Up to this point all of the force correction algorithms we have described only take into account the nodes directly adjacent to the effected nodes. This algorithm takes into account the nodes undergoing localized control forces whereas the previous algorithms were only intended to smooth the side effects of bending and twisting in adjacent nodes. Easing is a common technique used in animations and can produce much more life-like simulations. For example, an animated character that reaches up with their hand to scratch the top of their head does not typically do so in a flat accelerated rate. Rather the motion starts off slow, moves to a steady velocity, and then finally slows again as the hand approaches the target destination. In this sense the beginning of the motion is considered easing in – a gradual increase in force or velocity. And the end is considered easing out – a gradual decrease in force or velocity.

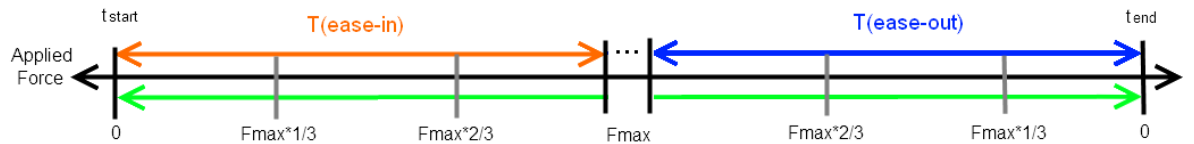
The same concept is applied to the soft body simulator however it has a different affect when applied to a localized metaphor. For instance, if we want to gently rotate the

end of a structure for a specified number of degrees then roll it back to the rest configuration we cannot simply apply force until the target is reached and then release all the controlling forces at once. Without easing force controls the twisting motion would appear to be jerky and could cause some osculation as the object returns to its resting configuration. Instead, the concept here is to enable the user to define how long the ease-in and ease-out forces should take to reach their peaks, so that there is not an instantaneous increase or decrease in velocity of the effected nodes.

To implement this we have added common controls to the base metaphor that will intercept the maximum applied force and then change its ratio as any metaphor applies a localized control. For a smooth motion we divide the maximum applied force by the specified time units for the easing effect and then accumulate the easing force at each time step until we reach the maximum force.

$$\frac{F_{max}}{t_{ease}} = F_{inc} \quad \text{Force accumulator} = \sum_{t_{start}}^{t_{ease}} F_{inc}$$

The configuration parameters are tunable to the total length of time that the metaphor is to be applied and will allow the user to ramp up (or down) the applied forces across a specific time range.



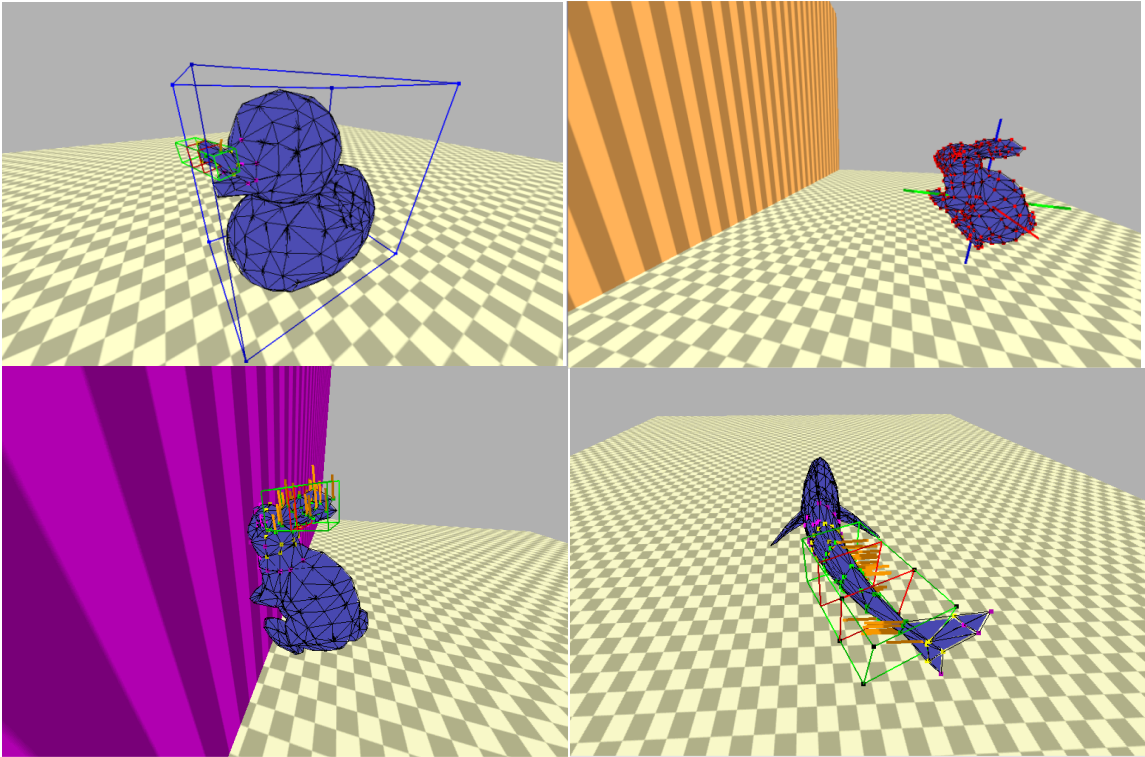
**Figure IV.9- Ease Timeline.**

So if a slow bend and slow release is desired the animator would divide the time range in half and apply ease-in and ease-out for each of the halves. Increments of force are directly related to how long the easing takes place.

## CHAPTER

### V. RESULTS AND DISCUSSION

#### Experimentation



**Figure V.1- Sample Experimentation.**

Each of the images above is a snippet from a localized control experiment.

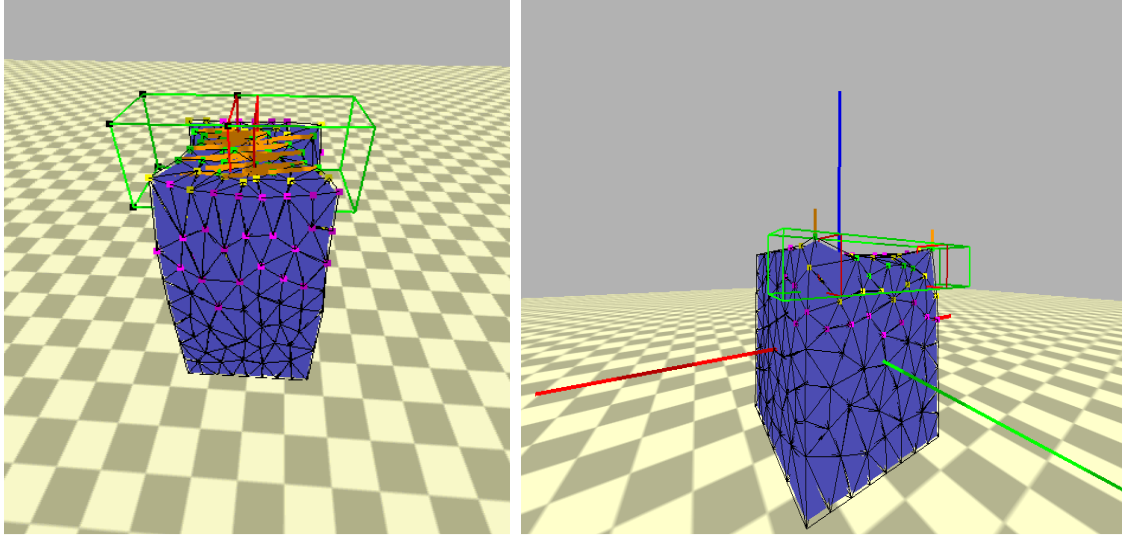
Through experimentation and testing we have hoped to demonstrate some of the unique capabilities of this implementation but do recognize that there are far more possibilities than what has been done here. A few of the most influential test cases involve objects moving from a resting state, fluid motion, bouncing, and even some collisions with immovable objects. At its core, the simulator is designed to capture live



physics state and utilize the physics projections making localized controls far more useful with moving objects than resting structures. Another interesting observation is that animals and living meshes seem to fit the criteria of objects that may require localized control more often than inanimate objects. This certainly is not a limitation of the framework but rather a well defined use case.

### **Geometric Structures and Resting State**

In this experimentation, geometric structures ultimately serve as good resources for demonstration purposes but tend to lack enough complex characteristics to fit the need for localized control. For example, the bend widget has the ability to create a parabolic bend by applying a common force to two ends of a bounding structure and an opposing force in the middle. This is not easily demonstrated with the bunny ear due to its flimsy properties but with a solid cube we can demonstrate with one edge how the tool operates. The same can be said for the spherical shaped widget since the rounding of edges can more easily be seen on the corner of a simple box like structure. In addition, we have found that the resting state falls more into a different category of research that has already been studied for localized control in other aspects. For these reasons the reader will see many geometric examples for the purposes of demonstration but should be aware that the implementation is far more powerful.



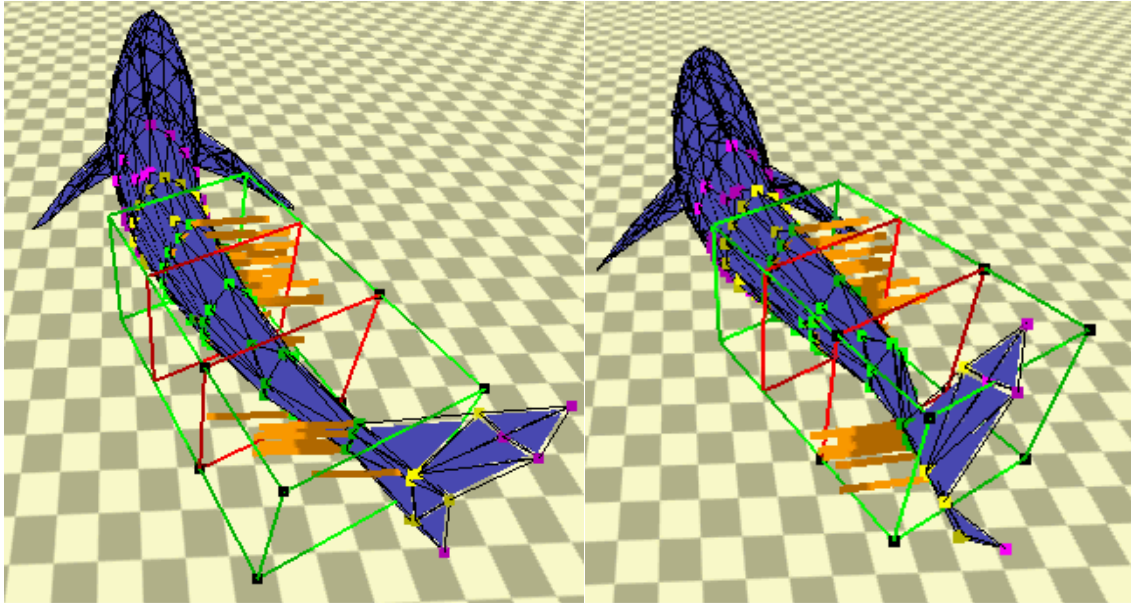
**Figure V.2- Geometric Structures.**

Simple geometric structures being deformed under the control fo the bend widget.

### **Moving Objects and Time Constraints**

An exemplary model that can benefit from localized control is that of the shark tail moving through a zero gravity environment. We start with a semi complicated mesh which has a dense core structure and tapers off into a fin, and then we try to control certain aspects of that localized area. We could place levers and pulleys to simulate a more mechanical affect however this is not required when we use our localized controls to replicate a swimming motion. To accomplish this we add velocity to the shark so that it is coasting through a weightless environment at a steady rate. Note that this simulation does not incorporate wind resistance or any sort of viscosity in the surrounding atmosphere so there is unrestricted motion in all direction. Once we have dialed in the initial conditions for this simulation we restart the dynamics timeline and record physics state at a constant rate until we have reached our stopping time. With this recording, we now have the ability to scrub back and forth to the exact moment in the scene where we

would like to see the shark flap a piece of its tail in a gentle manner. Adding the bend metaphor (in the stretch/compress mode) one can adjust the bounding structure to encompass the required portion of the tail and finally tweak the projected force directions.



**Figure V.3- Swimming Shark Experiment.**

The simulation is then started with localized control adjusted to the required effected nodes and tested for visual acceptance. Without error correction or air resistance the shark moves off course of the intended motion and needs refinement to correct this behavior. In addition, the tail moves in a manner too violent to produce the final intentions. The first attempt to correct this behavior is adjusting force magnitude and applying force correction techniques to try and smooth out the simulation. However when applying the force dampening method in this zero gravity environment it is observed that the inertia of the moving object is slowed from its steady rate. The overall

look of the swimming motion is improved due to the less violent reaction but this is still not a perfect motion. We still have two behaviors to correct; the slowed forward momentum of the shark and the tail snap back when the localized controls have reached the end of their time application.

The next step in this experiment is to apply only mitigating forces which will not directly dampen the existing motion and act more directly on mitigating the applied forces. This also improves the simulation much the same way the original force dampening did but when watching in real time the inertia is not slowed as much due to this force correction. This is more ideal for this type of simulation so this force correction tool will be used in the final production and we will move onto the snapping when localized forces are released. The ease-in and ease-out tools were designed to resolve these details and can be leveraged in a multitude of ways. For this simulation a simple 50/50 split of the applied force time interval seems to alleviate much of the snapping behavior introduced by an instantaneous force application. With this configuration the shark is now able to maintain a stable inertia and observes a smooth stroke to the left and slow release back to its resting state.

### **Limitations**

The graphics world has been benefited greatly by dynamic simulations thanks to their ability to automate the creation of animations that closely mimic physically accurate events. Currently there is no direct mapping between the real world and simulated world so we continue to impose heuristics to try and achieve realistic animations. This forces us to perform a high degree of trial and error testing to achieve our target simulations regardless of what methods we are using. Our methods are also subject to this type of

trial and error testing even though we have decoupled the need for material property configuration from intended behavior. When an animator simulates a scene and uses the timeline scrubber to obtain the temporal encoding we still expect them to apply metaphors and change many of the force correction algorithms until they are satisfied. Future works may continue to improve this by decreasing the trial and error; however artistic designs will always take time and multiple iterations to complete.

## **CHAPTER**

### **VI. FUTURE WORK**

Looking forward we see many new ideas that could be expanded upon from the core of this research and we also see great potential in its relevance for the future of deformable objects. The technologies that enable better human interactions are coming at a furious rate and as improved enabling technologies are introduced the demand for more intuitive programming is eminent. There are many facets of work that simply were not within the time constraints of this project however we set forth the challenge for ourselves and others interested in this topic to continue with some of the work described here.

#### **Force Correction**

Although we did derive smooth localized deformations, we saw room for a more data driven force correction algorithm. We originally set out to create error corrections algorithms that were able to make runtime decisions based on linkage strain and adjacency positions, unfortunately this will have to come in future work as time limitations prohibited the full implementation of such algorithms. During testing and implementation it was noted that a key factor to force correction could come from the direct measure of strain. In fact, since the simulator is adding localized forces to the body, node linkages can easily be severed if the user is not cautious of the forces they are applying. We would like to see more autonomous error correction that does not require

the animator to have detailed knowledge of the control forces and ultimately prevent this undesired behavior from occurring.

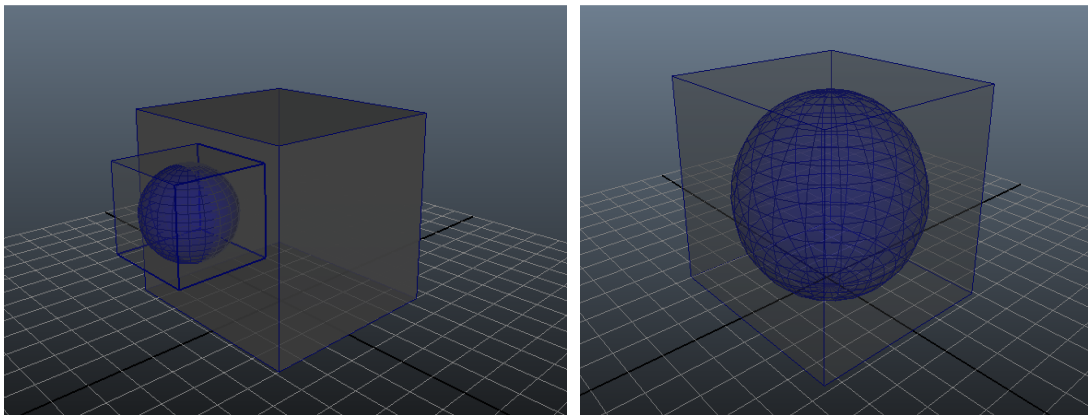
### **Temporal Material Configuration**

An interesting thought is to allow a soft body moving through a time continuum to change specific material properties to match configuration requirements. Furthermore, the ability to adjust the material properties of a deformable structure in a localized manor could lead to some intriguing simulations. The focus of the *Soft Body Simulator* was to stay in the force domain of the physics engine however we often observed desirable behaviors that mimicked changing material properties. Often a localized force interacts with different structures of the body and we apply more localized force here or there to meet a final goal when perhaps we could have manipulated a specific patch of the body to be more giving or rigid. We see that this is quite similar to the localized force control on an object and could be subject to similar error correction techniques that ultimately lead to an animator's rendering goal.

### **Example Based Implementation**

The example based manifold was one of the most desired future works that came from this experimentation and should be one of the first items tested. We see great potential in the ability for a user to artistically design final poses and configurations that can be directly embedded into the physics simulator. This could be the metaphor that supersedes all other metaphors since any of them can be derived from an artistic rendering and applied through the example based manifold. It may be found that this is too difficult to easily interact with so more user friendly metaphors similar to the sphere,

bend, or twist may still need to exist. Fortunately these could be reconstructed with the example manifold while leaving the door open for more advanced uses. This brings about other questions of how a localized metaphor truly works when being applied to the structure. Do we have final configurations for the entire structure but only apply it within the bounding structures? Or does the bounding structure scale to the final configuration so that it fits within localized region but is applied on a smaller scale.



**Figure VI.1- Localized Regions.**

### **User Interfaces**

There always seems to be room for improvement with user interfaces and this application is certainly no exception to that rule. As the framework evolves, we envision more visualization and hands on tools that can allow for better interpretation and more efficient development. A possible immediate enhancement could be to utilize gesturing with an input device other than the mouse and keyboard. Specifically, what if the iPad or other hand held touch screen device was given to an animator and allowed them to pick and pull at a deformable object with their finger tips. The artist could potentially draw



the target deformation over a timeline and the tools underneath the hood of the simulator would optimize constraints and simulate with automatic error correction fixing minor issues that the animator does not have to worry about.

### **Augmented Reality Data Collection**

Here we exploit the existing technologies of computer vision in the augmented reality area to construct a means for collecting user data. Humans have been interacting with computers for decades, yet we still lack ways to intuitively relay our thoughts and intentions into them. We envision the construction of an application, and possibly in conjunction with a physical hand held device, which can continually update a display with the gestures of human hands on a synthesized object. AR (Augmented Reality) technologies exist today which can read in video devices and detect positional coordinates of objects providing the foundation for this type of application. Using the video display we can project the users live environment on the monitor and overlay this image with an object or mesh that can be manipulated by hand gestures. So as the user looks at the display they can “push”, “poke”, “squash”, or move control points and watch how the object deforms in real time. As the user perturbs the object we can record their intended motions or how the deformation should take place allowing them to mold the object, somewhat analogous to clay.

In addition to a purely visual input feedback, an application could be built around a device that can collect the user’s physical touch. General motion and force could be detected as input so the device’s construction and shape is not completely essential to collect meaningful data. Suppose a force is applied in an upward curling motion to one of the corners, then a display is immediately updated with a corresponding deformation

correlating with that gesture. This is a different type of interaction with the program that gives the human user, perhaps, a more realistic or at least tangible experience.

### **Brain-Computer Interface**

Technologies that can read the human brain have been studied by neuroscientists for many years with significant progress in reading brainwaves and thoughts. There currently is research surrounding chip implants which physical penetrate the cerebral cortex providing a pipeline for reading neural firings. In conjunction to the chip there are devices which are placed against the scalp to read neural activity (electroencephalograph) [34]. This type of interaction with the brain can open a completely new set of applications that could benefit human prostheses control or non verbal communications. We also see this as a gateway to new computer visualization technologies. With these tools we can construct an application, similar to the augmented reality data collection, which can utilize the direct intentions of a human. Two way communications is provided by updating a display with the directives received from the brain applied to a virtually deformable object. In this manner, the application is able to receive the exact intentions of the artist and record invaluable information for the simulation.

### **Validation**

Intuitive design and usability is of the utmost importance as future and existing applications are developed. It will be essential to elicit constant user feedback and visual validation of simulations as new enabling technologies and data input tools are developed. We see two possible approaches for validation. First, distribute the frameworks and components that are constructed as public open source code for other's

to verify and employ in their own applications. This will bring in immediate feedback as well as sparking interest in new related works potentially spanning into fields of study beyond computer science.

The second aspect of validation is to bring in experts and researchers in human computer interaction and psychology to study the behavior of test subject using these new applications. This validation will begin with a survey of subjects ranging from many different skill sets and age groups to best understand what design improvements can be made. Starting with a set of undergraduate or graduate students in the computer science department should give a fundamental understanding of the usability of these tools for a relatively well trained group of subjects. Once the study of that test group is completed it may be illuminating to bring in younger children to test the intuitive design of the applications. Children and youth typically have had longer exposure to personal computers and thus can provide better feedback to how flexible and instinctive the user interfaces are.

Finally we would like to see a performance analysis that verifies what level of real time simulations can be achieve with these techniques as well as visual analysis of the deformations. Our small scale tests with simple meshes have shown that this can be a real-time framework for simple deformable bodies but can it be ported to medical simulators and potentially detailed gamming platforms?

The potential of dynamically simulated deformable bodies is quite vast and we foresee a great deal of intellectual wealth that can be gained by continuing these studies.

## REFERENCES

- [1] Adams, B., Ovsjanikov, M., Wand, M., Seidel, H.-peter, & Guibas, L. J. (2008). Meshless Modeling of Deformable Shapes and their Motion.
- [2] Allard, J, Cotin, S., Faure, F, Bensoussan, P.-J., Poyer, F., Duriez, C., Delingette, H., et al. (2007). SOFA--an open source framework for medical simulation. *Studies in health technology and informatics*, 125, 13-8. Retrieved from <http://www.ncbi.nlm.nih.gov/pubmed/17377224>
- [3] Barbič, J., & James, D. L. (2005). Real-Time subspace integration for St. Venant-Kirchhoff deformable models. *ACM Transactions on Graphics*, 24(3), 982. doi:10.1145/1073204.1073300
- [4] Bickel, B., Bäcker, M., Otaduy, M. a, Matusik, W., Pfister, H., & Gross, Markus. (2009). Capture and modeling of non-linear heterogeneous soft tissue. *ACM Transactions on Graphics*, 28(3), 1. doi:10.1145/1531326.1531395
- [5] Debunne, G., Desbrun, M., Cani, M.-P., & Barr, A. H. (2001). Dynamic real-time deformations using space & time adaptive sampling. *Proceedings of the 28th annual conference on Computer graphics and interactive techniques - SIGGRAPH '01*, 31-36. New York, New York, USA: ACM Press. doi:10.1145/383259.383262
- [6] English, E., & Bridson, R. (2008). Animating developable surfaces using nonconforming elements. *ACM Transactions on Graphics*, 27(3), 1. doi:10.1145/1360612.1360665
- [7] Faure, François, Barbier, S., Allard, Jérémie, & Falipou, F. (2008). Image-based Collision Detection and Response between Arbitrary Volume Objects. *Response, i*.
- [8] Faure, François, Gilles, B., Bousquet, G., & Pai, D. (2011). Sparse Meshless Models of Complex Deformable Solids. *ACM Trans. Graph.*, 30(4).
- [9] Fattal, R., and Lischinski, D. 2004. "Target-driven smoke animation" In Proc. of SIGGRAPH 2004, ACM Press / ACM SIGGRAPH, Comput. Graphics Proc., 441-448.
- [10] Gain, J., & Bechmann, D. (2008). A survey of spatial deformation from a user-centered perspective. *ACM Transactions on Graphics*, 27(4), 1-21. doi:10.1145/1409625.1409629
- [11] Gibson, S. F. F., & Mirtich, B. (1997). A Survey of Deformable Modeling in Computer Graphics. *October*.

- [12] Gilles, B., Bousquet, G., Faure, François, & Pai, D. (2011). Frame-based Elastic Models. *ACM Trans. Graph.*, 30(2).
- [13] Han, D.-P., Wei, Q., & Li, Z.-X. (2008). Kinematic control of free rigid bodies using dual quaternions. *International Journal of Automation and Computing*, 5(3), 319-324. doi:10.1007/s11633-008-0319-1
- [14] Irving, G., Teran, J., & Fedkiw, R. (2004). Invertible finite elements for robust simulation of large deformation. *Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation - SCA '04*, 131. New York, New York, USA: ACM Press. doi:10.1145/1028523.1028541
- [15] Jeon, H., & Choi, M.-hyung. (n.d.). Interactive Simulation of Motion Editing and Pattern-Based Control for Deformable Objects. *Control*, 1-10.
- [16] Kaufmann, P., Martin, S., Botsch, M., & Gross, Markus. (2009). Flexible simulation of deformable models using discontinuous Galerkin FEM. *Graphical Models*, 71(4), 153-167. doi:10.1016/j.gmod.2009.02.002
- [17] Kavan, L., Collins, S., Žára, J., & O'Sullivan, C. (2007). Skinning with dual quaternions. *Proceedings of the 2007 symposium on Interactive 3D graphics and games - I3D '07*, 39. New York, New York, USA: ACM Press. doi:10.1145/1230100.1230107
- [18] Kondo, R., Kanai, T., & Anjyo, K.-ichi. (2005). Directable animation of elastic objects. *Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation - SCA '05*, 127. New York, New York, USA: ACM Press. doi:10.1145/1073368.1073385
- [19] Kry, P. G. (2010). Volume Contact Constraints at Arbitrary Resolution. *Processing*, C(3), 1-10.
- [20] Magnenat-thalmann, N., & Iro, H. E. C. (n.d.). Joint-Dependentlocal Deformations for Hand Animation and Object Grasping.
- [21] Martin, S. (2011). Unified Simulation of Elastic Rods, Shells and Solids: Implementation Notes. *Computer*, (721), 1-3.
- [22] Martin, S., Kaufmann, P., Botsch, M., Wicke, M., & Gross, Markus. (2008). Polyhedral Finite Elements Using Harmonic Basis Functions. *Computer Graphics Forum*, 27(5), 1521-1529. doi:10.1111/j.1467-8659.2008.01293.x
- [23] Martin, S., Thomaszewski, B., Grinspun, E., & Gross, Markus. (2011). Example-based elastic materials. *ACM SIGGRAPH 2011 papers on - SIGGRAPH '11* (Vol.

30, p. 1). New York, New York, USA: ACM Press.  
doi:10.1145/1964921.1964967

- [24] Meier, U., López, O., Monserrat, C., Juan, M. C., & Alcañiz, M. (2005). Real-time deformable models for surgery simulation: a survey. *Computer methods and programs in biomedicine*, 77(3), 183-97. doi:10.1016/j.cmpb.2004.11.002
- [25] Methods, M., & Methods, M. (2004). Classification and Overview of Meshfree Methods. *Integration The Vlsi Journal*.
- [26] Mezger, J., Thomaszewski, B., Pabst, S., & Straßer, W. (2009). Interactive physically-based shape editing. *Computer Aided Geometric Design*, 26(6), 680-694. doi:10.1016/j.cagd.2008.09.009
- [27] Müller, M., Heidelberger, Bruno, Teschner, Matthias, & Gross, Markus. (2005). Meshless deformations based on shape matching. *ACM Transactions on Graphics*, 24(3), 471. doi:10.1145/1073204.1073216
- [28] Müller, M., Keiser, R., Nealen, a, Pauly, M., Gross, M., & Alexa, M. (2004). Point based animation of elastic, plastic and melting objects. *Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation - SCA '04*, 141. New York, New York, USA: ACM Press.  
doi:10.1145/1028523.1028542
- [29] Nealen, A., Müller, M., Keiser, Richard, Boxerman, E., & Carlson, M. (2006). Physically Based Deformable Models in Computer Graphics. *Computer Graphics Forum*, 25(4), 809-836. doi:10.1111/j.1467-8659.2006.01000.x
- [30] Nesme, M., Kry, P. G., Jeřábková, L., & Faure, François. (2009). Preserving topology and elasticity for embedded deformable models. *ACM Transactions on Graphics*, 28(3), 1. doi:10.1145/1531326.1531358
- [31] Ng, S.-pui, Ng, R., & Yu, W. (2005). Bilinear Approximation of Anisotropic Stress-Strain Properties of Woven Fabrics, 9(4).
- [32] O'Brien, J. F., Bargteil, A. W., & Hodgins, J. K. (2002). Graphical modeling and animation of ductile fracture. *ACM Transactions on Graphics*, 21(3), 291-294. doi:10.1145/566654.566579
- [33] Picinbono, G. (2003). Non-linear anisotropic elasticity for real-time surgery simulation. *Graphical Models*, 65(5), 305-321. doi:10.1016/S1524-0703(03)00045-6
- [34] Pfurtscheller G, Electroencephalography, Basic Principles, Clinical Application and Related Fields, E. Niedermeyer and F.L. Da Silva, Eds., Williams and Wilkins, Baltimore, Md, USA, 4th edition, 1998.

- [35] Popović, J., Seitz, S. M., Erdmann, M., Popović, Z., & Witkin, A. (2000). Interactive manipulation of rigid body simulations. *Proceedings of the 27th annual conference on Computer graphics and interactive techniques - SIGGRAPH '00*, 209-217. New York, New York, USA: ACM Press. doi:10.1145/344779.344880
- [36] Popovic, J., Seitz, S. M., and Erdmann, M. 2003. "Motion sketching for control of rigid body simulations" *ACM Transactions on Graphics*, Vol. 22, No. 4, 1034-1054.
- [37] Rodriguez, S., Lien, J. M., and Amato, N. M. 2006. "Planning Motion in Completely Deformable Environments" In *Proc. of IEEE Int. Conf. Robotics and Automation*, 2466-2471.
- [38] Sorkine, O., & Alexa, Marc. (2007). As-Rigid-As-Possible Surface Modeling. *Processing*.
- [39] Terzopoulos, D. (1988). Modeling Inelastic Deformation: Viscoelasticity, Plasticity, Fracture. *Computer*, 22(4), 269-278.
- [40] Terzopoulos, D., Platt, J., Barr, A., & Fleischer, K. (1987). Elastically deformable models. *ACM SIGGRAPH Computer Graphics*, 21(4), 205-214. doi:10.1145/37402.37427
- [41] Teschner, M., Heidelberger, B., Muller, M., & Gross, M. (n.d.). A versatile and robust model for geometrically complex deformable solids. *Proceedings Computer Graphics International, 2004.*, 312-319. Ieee. doi:10.1109/CGI.2004.1309227
- [42] Wang, H., & Brien, J. F. O. (2011). Data-Driven Elastic Models for Cloth: Modeling and Measurement. *Measurement*, 30(4).
- [43] Wang, H., & Brien, J. O. (2010). Multi-Resolution Isotropic Strain Limiting. *English*, 1-10.
- [44] Weber, O., Sorkine, O., Lipman, Y., & Gotsman, C. (2007). Context-Aware Skeletal Shape Deformation. *Computer Graphics Forum*, 26(3), 265-274. doi:10.1111/j.1467-8659.2007.01048.x
- [45] Wei, L.-yi, & Turk, G. (2009). State of the Art in Example-based Texture Synthesis. *Synthesis*, (Section 2).
- [46] Wirth, B., Bar, L., Rumpf, M., & Sapiro, G. (n.d.). A Continuum Mechanical Approach to Geodesics in Shape Space. *Computer*, 1-38.

- [47] Witkin, A., & Kass, M. (1988). Spacetime constraints. *ACM SIGGRAPH Computer Graphics*, 22(4), 159-168. doi:10.1145/378456.378507
- [48] Wojtan, C., Thürey, N., Gross, Markus, & Turk, G. (2009). Deforming meshes that split and merge. *ACM Transactions on Graphics*, 28(3), 1. doi:10.1145/1531326.1531382
- [49] Zhao, J., Wei, Y., Zhu, D., Xia, S., & Wang, Z. (2011). Path Tracking with Time Constraints: A Convex Optimization Approach. *Society*.